# Multi-Objective Landscape Analysis and Feature-based Algorithm Selection

Raphaël Cosson

**Université de Lille**
École doctorale **MADIS**

Laboratoire **Inria-Lille**
Unité de recherche **CRIStAL**

---

# Multi-Objective Landscape Analysis and Feature-based Algorithm Selection

Analyse de Paysage Multi-objectif
et Sélection Automatique d'Algorithmes

---

*Thèse présentée par*

**Raphaël Cosson**

En vue de l'obtention du grade de docteur de l'Université de Lille
Discipline : **Informatique et Applications**

**Soutenu le 20 décembre 2023 devant le jury composé de :**

| | | |
|---|---|---|
| Rapporteurs | **Carola Doerr** | Directrice de Recherche<br>Sorbonne Université, CNRS |
| | **Frédéric Saubion** | Professeur<br>Université d'Angers |
| Examinateurs | **Jean-Christophe Routier** | Professeur, Président du jury<br>Université de Lille |
| | **Sébastien Vérel** | Professeur<br>Université du Littoral Côte d'Opale |
| Directeur | **Bilel Derbel** | Professeur<br>Université de Lille |
| Encadrant | **Arnaud Liefooghe** | Maître de Conférence<br>Université de Lille |

---

CRIStAL
Centre de Recherche en Informatique, Signal et Automatique de Lille

Inria

Université de Lille

# Résumé

Cette thèse porte sur l'analyse de paysage de problèmes d'optimisation combinatoires multi-objectifs. La résolution de tels problèmes est une tâche difficile, particulièrement en optimisation multi-objectif en raison de la nature contradictoire des objectifs. Ces situations apparaissent fréquemment dans de nombreux scénarios réels et constituent un véritable défi pour les algorithmes. Les approches de résolution reposent sur la découverte de solutions qui forment des compromis intéressants. Parmi ces approches, les algorithmes évolutionnaires s'avèrent particulièrement adaptés à leur résolution. Cependant, il existe de nombreux algorithmes évolutionnaires et leur performance varie en fonction du problème à résoudre.

Dans cette thèse, nous cherchons à comprendre la raison de ces variations et à déterminer l'algorithme le plus intéressant pour un problème donné. Pour cela, nous nous intéressons particulièrement à l'étude des caractéristiques internes aux instances de problèmes. Cette étude porte sur l'analyse de paysage de problèmes d'optimisation multi-objectifs. L'analyse de paysage permet de caractériser les structures locales internes à un problème d'optimisation. L'intérêt est à la fois fondamental, pour mieux comprendre les difficultés des algorithmes. De plus, elle offre un intérêt pour l'automatisation de la sélection d'algorithmes. Cet aspect pratique est tout particulièrement important, puisqu'il permet de choisir l'algorithme le plus performant pour un problème non rencontré auparavant.

Dans un premier temps, nous proposons une approche d'analyse de paysage par décomposition de problèmes multi-objectifs. Cette approche est alors étudiée sur une collection de problèmes d'optimisation aux caractéristiques connues. L'approche est ensuite appliquée expérimentalement pour sélectionner automatiquement le meilleur algorithme pour un problème donné. Dans un troisième temps, les investigations précédentes sont approfondies, notamment sur le coût de l'analyse du paysage et sa prise en compte dans le modèle de sélection. Enfin, une nouvelle collection de problèmes combinatoires multi-objectifs est proposée. Elle apporte un nouveau critère de difficulté pour les algorithmes : l'hétérogénéité entre les objectifs. Nous montrons que cette nouvelle propriété est observable par le biais de l'analyse de paysage par décomposition.



**Mots-clés** Optimisation Combinatoire, optimisation multi-objectif, Algorithmes évolutionnaires, Analyse de paysage, Sélection d'algorithmes.

# Abstract

This thesis focuses on landscape analysis for multi-objective combinatorial optimization problems. Solving such problems is a difficult task, especially in a multi-objective setting, due to the conflicting nature of the considered objectives. Moreover, in a black-box setting, no knowledge about the problem can be assumed, and one can only one probe the objective functions to evaluate the quality of solutions. In such a setting, evolutionary algorithms constitute a popular solving approach. However, one can find different evolutionary algorithms, with different components and parameters, leading to a different performance depending on the problem being solved.

Understanding the difference in performance and determining the most interesting algorithm (or algorithm component) for a given problem instance requires studying its intrinsic characteristics. In this context, fitness landscape analysis has a fundamental interest as it helps to gain a fundamental understanding of what makes a problem difficult to solve. In addition, it offers new opportunities for automated algorithm selection, when one has to decide the best algorithm to execute for an unseen problem.

In this thesis, we propose a new landscape analysis methodology using the decomposition paradigm for multi-objective problems. We study this approach for a set of optimization problems with known characteristics. The method is subsequently investigated on more complex tasks, in particular for solving the algorithm selection problem. Then, we push our experiments further with a study on the cost of the landscape analysis itself. Further cost reduction is achieved by modifying the sampling methods necessary for landscape analysis while maintaining a degree of robustness of the proposed landscape features. Finally, a new collection of multi-objective combinatorial optimization problems is proposed, bringing a new challenge for algorithms by including heterogeneity between the objectives. We show that this property is observable through decomposition-based landscape analysis.



**Keywords:** Combinatorial Optimization, Multi-objective Optimization, Evolutionary Algorithms, Landscape Analysis, Automated Algorithm Selection.

# Acknowledgements

I would like to express my deepest appreciation to the people the people who contributed, more or less directly, in the realization of this PhD.

I would like to thank first Carola Doerr and Frédéric Saubion, for having accepted to be the reviewers of this PhD. I would also like to thank Jean-Christophe Routier, the chair of my PhD jury and Sébastien Vérel, for having accepted to participate in the jury as well. I would like to express all my gratitude for the constructive discussions during my PhD defense.

I would like to express my deepest thanks to my great PhD director Professor Bilel Derbel. Thank you for the opportunity that you gave me. I learned a lot from your advice, your positive remarks and your unwavering enthusiasm. I would also like to thank my great PhD supervisor Arnaud Liefooghe. In addition to your numerous scientific remarks, Your wise advice encouraged me to give my best to improve the quality of our research.

Carola Doerr, Thomas Weise, thank you for opening the door to me in this field of research and for passing on to me the passion that has driven me these years.

Finally, I thank my family, parents, and friends for their unconditional support and encouragement throughout these years.

# Contents

# List of Figures

# List of Tables

# Introduction

## Motivation

Multi-objective optimization problems require to optimize several objective functions simultaneously. One major difficulty inherent to multi-objective optimization is the presence of conflicts between the objective functions. In contrast to single-objective optimization, where one has to compute one single optimal solution, multi-objective algorithms have to find a whole set of solutions that minimize or maximize all objectives simultaneously. This set of solutions is called the *Pareto set* or the *efficient set*. It constitutes the best trade-offs between the conflicting objectives. The computation of the Pareto set is not always feasible in a reasonable time. This is why a large body of the literature in multi-objective optimization aims at finding a high-quality approximation of the Pareto set.

Many algorithms have been proposed to approximate the Pareto set. Among them, evolutionary algorithms are based on evolving a population of solutions. These algorithms have numerous components and parameters. The actual efficiency of such algorithms varies with respect to the optimization problem. It has been shown in recent years that these variations in performance can be related to the internal structure of the optimization problems. This thesis is concerned with *fitness landscape analysis* as a tool allowing to better characterize the structure of an optimization problem and its relation to algorithm performance. Fitness landscape analysis uses the concept of solution neighborhood to define a multi-dimensional space. Then, the resulting landscape/space can be analyzed, and numerical values called *features* can be extracted in order to grasp the structure of a problem instance. In addition to the fundamental interest in understanding optimization problems, fitness landscape analysis highlights the difficulty of solving an optimization problem with a specific structure. By connecting the landscape features with the algorithm behavior, it is for instance possible to predict the expected performance of an algorithm. As such, it becomes possible to address the problem of automated algorithm

selection, which consists in deciding the best algorithm to select when solving a given optimization problem instance.

In this thesis, we focus on deriving new fitness landscape analysis techniques for multi-objective combinatorial optimization, as well as the design of an effective approach to solve the algorithm selection problem. The main lines of research considered in this thesis can be summarized as follows:

- **How to design multi-objective landscape features?**
  Most of the work on multi-objective landscape analysis is based on two concepts specific to multi-objective optimization, namely, dominance and indicators. In this thesis, we aim to push further the available tools by investigating the application of the so-called decomposition paradigm in the design of informative landscape features.

- **How to integrate landscape features for addressing high-level optimization tasks ?**
  The features obtained by fitness landscape analysis can be used to address the algorithm selection problem, which constitutes a challenging high-level optimization task. A typical machine-learning-inspired approach consists in building an empirical performance prediction model to capture the relation between the landscape and algorithm performance. This applies both for single-objective and multi-objective optimization problems. However, two difficulties are encountered: the importance of the features and the evaluation of the prediction model. In fact, features generally form of a large set of numerical values, and it is not always clear why a particular feature set is accurate. Moreover, features complement each others and their impact on a performance prediction model is to be analyzed.

- **What are the effects of the sampling strategies used for feature computation?**
  Extracting landscape features requires to sample a set of solutions. Different sampling approaches have been proposed. However, the value of the same feature can vary drastically depending on the sampling method used to compute it. Hence, it is crucially important to understand the impact of the sampling methods on feature computation cost and on feature accuracy.

- **How to reduce the computation cost in order to improve the quality of automated algorithm selection?**

Feature sampling cost can directly influence the performance of the considered prediction models. As the goal is to determine the most suitable algorithm for a given problem instance within a given budget, computing features prior to the algorithm execution impacts the overall available budget at the solving stage. The computation of features requires the evaluation of a sample of solutions. The larger the sample, the more accurate the features. However, the most accurate features may not be useful if they require a large amount of budget. Conversely, reducing the cost may reduce the model prediction quality, and the prediction accuracy become questionable. To address this issue, we investigate a metric to compare the efficiency of prediction models and we propose to analyze the impact of the cost of computing features on their efficiency and accuracy.

- **How to construct benchmark problems with heterogeneous objectives in terms of their individual difficulty, and how heterogeneity impacts the landscape ?**
  Among the difficulties inherent to multi-objective optimization, the heterogeneity between objectives is an important issue. Heterogeneity can take many aspects. One aspect is the difference in difficulty between the objectives in terms of ruggedness and multi-modality, which has not been explicitly explored so far. In this thesis, we introduce heterogeneous multi-objective problems and we carefully examine the impact of heterogeneity on both the landscape properties and algorithm performance.

# Overview of Contributions

Existing investigations on multi-objective landscape analysis are related to two main concepts: the dominance relation and the use of quality indicator. As it will be detailed later, dominance introduces a partial order among solutions. Quality indicators are used to compare two sets of solutions and to assess their performance. These two concepts are at the heart of several existing landscape-oriented approaches. In this thesis, we propose to investigate the design of new landscape approaches based on the concept of decomposition. This is in fact at the heart of different state-of-art multi-objective algorithm such as MOEA/D. Therefore, our first contribution is an approach using the decomposition paradigm to compute metrics characterizing the landscape. The proposed approach decomposes the objective space using a set of weight vectors to generate a set of single-objective sub-problems. As the multi-

objective problem is transformed into a set of single-objective sub-problems, single-objective features can be computed for each sub-problem. Then, the so-obtained single-objective sub-problems landscape features are aggregated into multi-objective features. This approach is empirically analyzed from two perspectives, before and after the aggregation, on a collection of bi-objective benchmark problems with known properties.

In our second contribution, three approaches for multi-objective landscape analysis, respectively indicator-based, dominance-based and decomposition-based, are gathered to address two high-level tasks: predicting benchmark parameters and automatically selecting the best algorithm for a given problem instance. Each of these tasks relies on the use of machine learning models. In the first task, the model uses the landscape features to predict the benchmark parameters. In the second task, the features are used to predict the expected performance of a set of algorithms. The first task highlights the ability of the features in characterizing and classifying problem instances based on their respective landscape characteristics. The second task is an approach to solve the algorithm selection problem. In this task, the machine learning model uses algorithm performance on a set of problem instances to learn which algorithm is expected to be the most efficient at a fixed budget. In this second contribution, both tasks are analyzed through extensive experiments, showing the importance of the considered feature sets and the synergy between them.

The third contribution aims at deepening our understanding of landscape features. In fact, in the previously mentioned high-level tasks, the feature computation costs are ignored in order to better study their accuracy. However, in practice, feature cost can be of critical importance. The feature cost impacts both the features quality and the underlying algorithm selection approaches. Generally speaking, the larger the solution sample used for feature value extraction, the more accurate the features. In our setting, algorithm selection methods aim at predicting the most effective algorithm in order to have the best approximation quality for a given fixed budget. Hence, feature computation cost can negatively impact the efficiency of the considered prediction models. As such, our third contribution includes the analysis of two sampling strategies to provide an improved control over the sample size, and the underlying feature cost. The impact of feature cost on algorithm selection is then analyzed in a fine-grained manner.

In the last contribution of this thesis, we define a new collection of multi-objective combinatorial optimization benchmark problems with heterogeneous objectives. The proposed benchmark is thoroughly investigated with a landscape analysis approach. The proposed benchmark considers objectives with a different level of difficulty. Heterogeneity can appear in several ways, such as

different computation times or different relative scaling between the objectives. In our case, heterogeneity is to be understood in terms of the differences in difficulty between the objectives. We report a comprehensive fitness landscape analysis using the proposed heterogeneous benchmark for bi-objective problems. To observe how heterogeneity impacts the landscape, problems with a small number of variables are first fully enumerated, and then larger problems are analyzed using the previously considered landscape features. Besides, two state-of-the-art algorithms, namely NSGA-II and MOEA/D, are considered in order to study how heterogeneity impacts the performance and behavior of evolutionary algorithms.

# Outline

The document is organized as follows. In Chapter 1, some general concepts and definitions related to multi-objective optimization are provided. Different solving approaches are described, as well as the main optimization benchmark used in all our experiments. In Chapter 2, important notions related to fitness landscape analysis are described before introducing our first contribution. More specifically, using the decomposition paradigm, we design an approach to compute multi-objective features and we perform a first experimental analysis. In Chapter 3, the parameter prediction and automated algorithm selection tasks are introduced. The proposed decomposition-based landscape features, together with existing multi-objective features, are used to characterize black-box optimization problems and to predict the best out of three competing algorithms. In Chapter 4, the impact of the cost of the sampling method is analyzed using two cost-adjustable strategies, highlighting the stability of landscape features and their effect in the prediction tasks. In Chapter 5, we propose a new multi-objective benchmark with a different tunable difficulty for each objective, and we analyze the impact of such heterogeneity on the landscape properties. Finally, in Chapter 6, we summarize our main results and we discuss some perspectives.

# Chapter 1

# Background

In this chapter, we introduce the main concepts needed for understanding the different studies presented in the following chapters. First, multi-objective optimization is defined, together with the notion of dominance. Secondly, some solving methods are presented, with a particular focus on evolutionary algorithms. Three different multi-objective evolutionary algorithms are introduced, with the algorithmic details of two state-of-the-art algorithms used in the remaining of this thesis. Then, the fourth section explains the methods used to analyze the performance of multi-objective black-box optimization algorithms. Finally, $\rho$MNK-landscapes, a collection of multi-objective combinatorial benchmark problems, is defined and the motivations for using them in the scope of our work is highlighted.

## 1.1 Multi-objective Combinatorial Optimization

### 1.1.1 General Concepts

An optimization problem can be defined by two components: a search space $\mathcal{X}$ and an objective function $f$. The search space is the set of all solutions $x \in \mathcal{X}$ and the objective function $f : \mathcal{X} \rightarrow \mathbb{R}$ assigns an objective value to each solution. Among the possible solutions of a given problem, one has to compute the ones which maximize (or minimize) the objective function. For several optimization problems, the mathematical formulation of the problem is given in a generic form and depends on specific input parameters/data. For example, the well-known knapsack problem [72] needs a list of items with a weight and a value per item. The optimization *problem* refers to the generic formulation, whereas a *problem instance* refers to the problem under specific parameters/data.

The search space, also called decision space, is the set of all candidate solutions. Given a set of $n$ variables, where the number of variables $n$ is the problem dimension, the search space is typically composed of all possible combinations of variables values. The second component of an optimization problem is the objective function. The function assigns each solution with a numerical value called fitness, rendering the relative quality of the solution. Without loss of generality, we assume it is to be maximized. The formulation of the objective function of an optimization problem may not be known by the solver, and many often no properties about the function can be assumed. This results in settings called black-box, where algorithms have no particular knowledge about the objective function they need to solve. In other words, in a black box optimization setting, an algorithm can only call the objective function to evaluate the fitness of a solution.

The goal of an optimization algorithm is to obtain optimal solutions for a given problem as efficiently as possible. The most common optimization cost is the execution time of the algorithm. Each optimization problem belongs to a class of complexity. The complexity classes indicate the minimal cost required to solve any problem instance, as a function of the problem dimension. In the following, we focus on hard problems for which the optimal solution cannot be found in polynomial time. In the case of black-box problems, no specific assumptions on the computation time of the objective function can be made. Therefore, the optimization process seeks to minimize the number of calls to the objective function, i.e., the number of solutions evaluated.

### 1.1.2 Definitions

This thesis is concerned with multi-objective combinatorial optimization problems. A problem is said to be multi-objective if there exists more than one objective function to optimize simultaneously. Consequently, the fitness of each solution $x \in \mathcal{X}$ is a tuple of $M$ values such that $F(x) = (F_1(x), \ldots, F_M(x))$.

**Definition 1** ***: Multi-objective combinatorial optimization problem***

$$P = \begin{cases} maximize/minimize : F(x) = (F_1(x), \ldots, F_M(x)) \\ x \in \mathcal{X} \end{cases} \tag{1.1}$$

*Where $M$ is the number of objectives, $\mathcal{X}$ is the solution space, $x = (x_1, x_2, ..., x_n)$ is a solution, $F : \mathcal{X} \rightarrow \mathcal{Z}$ is the objective function vector and $F(x) = (F_1(x), ..., F_M(x))$ is the vector associated to $x$ in the objective space $\mathcal{Z} = \mathbb{R}^M$.*

Without loss of generality, we consider that all objective functions are to be maximized.

### 1.1.3 Dominance Relation and Pareto Optimum

The main difference between single-objective optimization and multi-objective optimization is the absence of total order between solutions. In fact, in the single-objective case, each solution $x$ in the search space $\mathcal{X}$ has a scalar objective value. For each pair of solutions $x, x' \in \mathcal{X}$, it is then possible to order solutions as illustrated in Figure 1.1: $x > x'$ or $x = x'$ or $x < x'$. In the case of multi-objective optimization, the order is not total. For example, Figure 1.2 illustrates three solutions of a bi-objective problem. The first objective $F_1$ actually corresponds to the objective of Figure 1.1. Considering only the first objective, the yellow solution has the highest objective value and the red has the lowest. However, this order is not preserved according to the second objective $F_2$. In a multi-objective problem, a solution can have a good quality with respect to one objective and the worst quality with respect to another objective. The objectives may be contradictory, such as for the yellow and the red solutions, which are the best solutions according to the objectives $F_1$ and $F_2$ respectively. It is unlikely that a solution maximize all objectives simultaneously. In the example, as both the red and yellow solutions are better than each other on one of the objectives, they are said to be incomparable. Conversely, the green solution has a smaller objective values according to both objectives in comparison to the yellow solution, it is therefore said that the yellow solution dominates the green solution. As such, there is no total order determining which solution is the best using the objective vector.

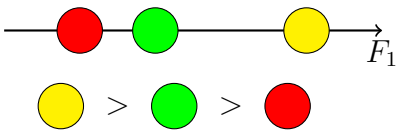


Figure 1.1: Total order for single-objective optimization.

The relationships determining incomparability and dominance came from the Pareto dominance relation and are formally defined below. Because of these relationships, we are not interested in a single solution but in a set of optimal solutions called the Pareto set.

**Definition 2 : *Pareto dominance***
*Given two objective vectors $z, z' \in \mathcal{Z}$, $z$ dominates $z'$ if $z_i \geq z'_i, \forall i \in \{1, ...M\}$*

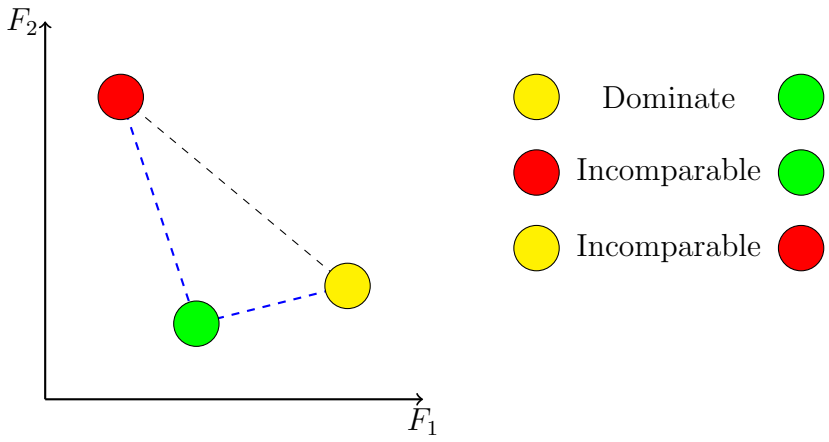


Figure 1.2: Lack of total order for multi-objective optimization. Red and yellow solutions are incomparable. The yellow solution dominates the green solution.

*and* $\exists j \in \{1, ...M\}$ *such that* $z_j > z'_j$. *A solution* $x \in \mathcal{X}$ *dominates* $x' \in \mathcal{X}$ *if the objective vector* $F(x)$ *dominates* $F(x')$.

**Definition 3 : *Incomparable solutions***
*Given two objective vectors* $z, z' \in \mathcal{Z}$, $z$ *and* $z'$ *are incomparable if* $\exists i, j \in \{1, ...M\}$ *such that* $z_i > z'_i$ *and* $z_j < z'_j$. *A solution* $x \in \mathcal{X}$ *is incomparable to* $x' \in \mathcal{X}$ *if the objective vectors* $F(x)$ *and* $F(x')$ *are incomparable.*

**Definition 4 : *Dominated solutions***
*An objective vector* $z \in \mathcal{Z}$ *is said to be dominated by the objective vector* $z' \in \mathbb{R}$ *if* $\forall i \in \{1, ...M\}, z_i \leq z'_i$ *and* $\exists i \in \{1, ...M\}$ *such that* $z_i < z'_i$. *A solution* $x \in \mathcal{X}$ *is dominated by* $x' \in \mathcal{X}$ *if the objective vector* $F(x)$ *is dominated by* $F(x')$.

**Definition 5 : *Pareto optimal solution***
*A solution* $x \in \mathcal{X}$ *is said to be Pareto optimal if* $\forall x' \in \mathcal{X}, x \neq x', x$ *is non-dominated by* $x'$.

**Definition 6 : *Pareto set***
*The Pareto set PS is the set of all non-dominated solutions*
$PS = \{x | x \in \mathcal{X}, x$ *is Pareto optimal*$\}$

**Definition 7 : *Pareto front***
*The Pareto front $PF$ is the projection of all non-dominated solutions into the objective space $PF = \{F(x) \in \mathcal{Z} | x \in PS\}$*

**Definition 8 : *Ideal point***
*The ideal point $z^* = (z_1^*, ..., z_M^*)$ is the vector which maximizes all objective functions independently, $z_i^* = max_{x \in \mathcal{X}} F_i(x), \forall i \in \{1, ..., M\}$*

**Definition 9 : *Nadir point***
*The nadir point $z^n = (z_1^n, ..., z_M^n)$ is the vector constituted by the worst values from the Pareto front, which minimizes all objective functions independently, $z_i^n = min_{x \in PF} F_i(x), \forall i \in \{1, ..., M\}$*

## 1.2 Solving Methods

There are mainly three strategies to solve multi-objective optimization problems. Among these, a human, denoted as the decision-maker, acts at different moments. The decision-maker knows the usefulness of the solutions once found and can therefore provide guidance to the search. The three strategies are:

- **A priori** methods. The decision-maker determines priorities among the objectives. An objective can be considered to be the most important. Others objectives can be set as constraints (positive or negative), meaning that the solution must reach a defined objective value which is not necessarily the optimum. Alternatively, objectives can be aggregated into a single-objective, such as with a weighted sum. The main advantage of this approach is the creation of a total order between solutions. Then, single-objective resolution methods can be applied to search the optimal solution. On the other hand, the main difficulty lies in the modelling of preferences between the objectives. A wrong characterization will return a solution which is not good for the decision-maker. Both steps, characterization and resolution, must be repeated to find a better solution.

- **A posteriori** methods. No distinction is made between the objectives. Therefore, all solutions from the Pareto set are possible candidates and must be found. In this case, the decision maker acts at the end, deciding among Pareto optimal solutions the one to use.

- **Interactive** methods. In this third strategy, the decision-maker and the resolution method cooperate to obtain a good solution. Depending on

intermediary results, the decision-maker refines the priorities between the objectives, enabling the search to be adapted according to their needs. Nevertheless, one constraint of this method is that the decision-maker is present at the different stages of the search.

The choice of the approach depends on the known properties of the problem. In this thesis, we only consider multi-objective algorithms belonging to the **posteriori** methods. In particular, we considered only the first step, which consist in finding the Pareto set.

### 1.2.1 Exact and Approximate Algorithms

One may consider two classes of algorithms when solving a optimization problem. On the one hand, exact algorithms seek the optimal solution or the set of Pareto optimal solutions in the context of multi-objective optimization. On the other hand, approximation algorithms seek to get as close as possible to the optimum, without necessary reaching it. Although the exact set of optimal solutions is preferable to an approximation, the complexity of optimization problems is often a barrier to exact algorithms. For combinatorial problems, designing an efficient exact algorithm is not always possible due to a number of factors such as the cost of evaluation, the intrinsic computational complexity of the problem, or the size of the search space. That is why in this thesis we are interested in approximation algorithms. Among the class of approximation algorithms, some guarantee a quality level called $k$-approximation, i.e., the quality of the computed solution is at most a $k$ multiplicative factor from the optimal one. Finding such a $k$-approximation is not always trivial nor possible, and it is often problem specific. Besides, such methods are not compatible with black-box optimization.

In this thesis, we are interested in heuristic search algorithms, which can be viewed as approximation algorithms that do not provide a theoretical guarantee on the solution quality ratio. Evolutionary algorithms are such an example, which form one of the main branches of meta-heuristics [13]. This family is widespread due to its versatility. In fact, it can be successfully applied to efficiently solve a wide range of optimization problems, and it constitutes a popular approach for tackling multi-objective optimization problems, which is the main concern of this thesis.

### 1.2.2 Evolutionary Algorithms

Evolutionary algorithms [29] are an important class of randomized/stochastic heuristic algorithms. Originally inspired by the evolution of living species,

these algorithms maintain a population of solutions which is used to generate new solutions called offspring. In preparation for the next iteration, the newly generated solutions are first evaluated. Then, a new population is selected among solutions from both the previous population and the generated offspring, and so on. A general schema of how an evolutionary algorithm work is presented in Figure 1.3.

**Initialization**
An evolutionary algorithm needs an initial population to start with. This population will be evolved in the subsequent iterations. Most of the time, the initial population is randomly generated. There exist alternative methods to generate an initial population, such as a Latin hypercube [43] for non-combinatorial problems or via problem-specific heuristics. The size of the population is a parameter that influences performance. The larger the population, the more diverse the solutions in the search space. Conversely, when the population is small, the algorithm typically performs, for a same budget, a greater number of iterations to improve the population. As a result, exploitation is favored instead of exploration. The initialization is usually done only at the beginning, however some heuristics may include a population restart during execution. The goal of the restart is to avoid a situation in which the algorithm stagnates when generations no longer produce interesting solutions.

**Selection**
Similar to the evolution of the living spices, the construction of new solutions is based on two steps: reproduction between members of the population and mutation. The selection step is to identify parent solutions from the population that will be combined in the reproduction step to form new offspring solutions. Selection operators choose among the population which parents are promising. These solutions are not necessarily those with the best objective values, because they can be local optima or solutions too regularly used in the selection. There are many selection operators, and they are often stochastic processes [103, 84, 45]. The most common/popular operators are:

- The roulette-wheel selection [60] randomly selects parents from the available solutions with a non-uniform distribution. The probability for an individual to be selected is based on the objective values of each solution.

- The stochastic universal sampling [70] is a variant of roulette-wheel limiting the risks of early convergence.

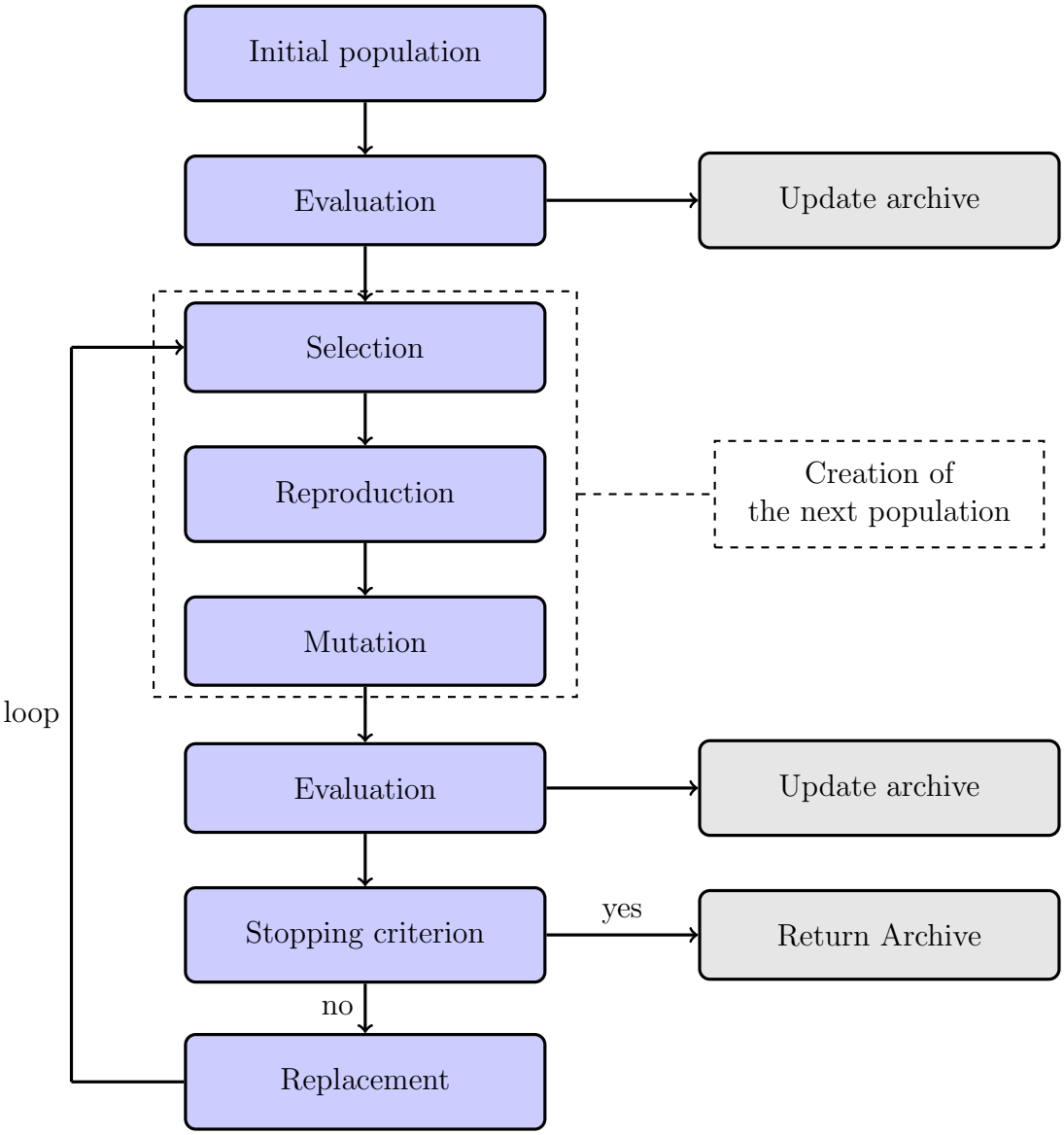


Figure 1.3: General diagram of a multi-objective evolutionary algorithm

- The linear selection [34] is also a variant of the roulette-wheel, using the ranking of solutions instead of their objective values to define the distribution of probabilities.
- The tournament selection [30] is a selection process which compares a pair of solutions until only one is left. It can be modeled as a binary tree. Each of the leaves represents a solution of the population. At each node, two solutions are compared, and the best one is selected for the next round.
- The random selection is, as the name suggest, a fully stochastic methods to selects parents.

Notice that, in all the above selection operators, at the exception to random selection, solutions need to be compared based on their fitness. In the case of multi-objective optimization, the dominance relation can be used instead of the fitness to compare solutions.

**Reproduction**

The reproduction step generates new solutions by recombination of parent solutions. The solutions generated are called offspring. There are several operators for reproduction [82, 85]. Two examples are given below.

- Single and multiple point crossover [25, 82]. For the single point crossover, a random point is selected among the variables. In the newly generated solution, all variable values before the point are taken from the first parent and all variable values after the crossover point are taken from the second parent. In the multiple point crossover, more than one point is selected, and the variable values are taken between the crossover points alternatively from the two parents.
- Uniform crossover [87]. Each variable of the offspring is selected at random from the parents.

**Mutation**

The mutation operator is a unary operator. It induces a variation in the variable values to generate a new solution. The goal of the mutation is to introduce additional diversity, and potentially to improve parent/offspring solutions. In the context of binary optimization problems, it is usual to use a mutation operator which flips each variable with a given probability. The probability is called the mutation rate. There are variants of this operator where the mutation rate is not fixed, but is instead set dynamically [27].

**Evaluation**
Once new solutions are generated, it is necessary to evaluate them. The evaluation is a call to the objective function having as input a given solution. The evaluation must be done before the replacement step. For multi-objective optimization, the evaluation is usually followed by an update of an *archive*. The archive usually keeps track of all non-dominated solutions found so-far. In the context of black-box optimization, the evaluation is considered to be the most time-consuming step. Thus, the cost of the algorithm is computed as a function of the number of calls to the objective function.

**Replacement**
The last step is to select among the current population and the offspring which solutions to keep as a new population for the next iteration of the algorithm. There are many approaches to replace the old population. When accepting solutions that are not necessarily better than the previous ones, the algorithm promotes exploration. On the contrary, keeping only the best solutions favors exploitation, which might be at the cost of diversity. Once the replacement step done, the algorithm repeats the previous steps until a stopping criterion is satisfied.

## 1.3 Multi-objective Evolutionary Algorithms

Most multi-objective evolutionary algorithms fall into one of three categories [104, 12, 26]. Dominance-based algorithms use the dominance relation as partial order to evolve solutions. Indicator-based algorithms try to improve a quality measure for comparing solutions. Decomposition-based algorithms manage to decompose the multi-objective problem into several single-objective sub-problems.

### 1.3.1 Decomposition-based Approaches

Among decomposition-based multi-objective evolutionary algorithms, the so-called MOEA/D algorithm [104] has received a lot of attention due to its ability to efficiently solve various problems. MOEA/D is based on the resolution of single-objective sub-problems, which are scalar aggregations of the initial objectives. Sub-problems are then solved by techniques dedicated to single-objective optimization, while allowing cooperation among sub-problems solving; so to improve the speed of convergence and the diversity of the population. The MOEA/D algorithm has been improved many times [62, 102, 86, 73].

The main steps of the algorithm are presented in Algorithm 1 and are discussed below.

**Algorithm 1** MOEA/D [104]

**Input:** $F = (F_1, ..., F_M) : \mathcal{X} \longrightarrow \mathcal{Z}$: fitness function
$g(x, w) \longrightarrow \mathbb{R}$: aggregation function.
$W = \{w_1, \ldots, w_\mu\}$: weight vectors.
$V$: Neighborhood size.
$C$: stopping condition.

▷ set the initial solution for each weight vectors
$P \leftarrow \{\textbf{random solution}, \forall i \in \{1, ..., \mu\}\}$
$S \leftarrow \{F(p), \forall p \in P\}$ ▷ Evaluate the population
$A \leftarrow$ Non-dominated$(S)$ ▷ Initialize archive
$z^* \leftarrow \{\max(F_i(s), \forall s \in A), \forall i \in \{1, ..., M\}\}$ ▷ Initialize the reference point
**repeat**
  **for** $i \in \{1, ..., \mu\}$ **do** ▷ for each sub-problem
    ▷ Extract the neighborhood of the current sub-problem
    $T \leftarrow \left\{(j, P[j]), \forall j \in \{1, ..., \mu\}, j \neq i \text{ such that } |i - j| < \frac{V}{2}\right\}$
    *Parents* $\leftarrow$ selection$(T)$ ▷ Select parents among the neighborhood
    *Offspring* $\leftarrow$ crossover(parents)
    $p_i \leftarrow$ mutate(offspring)
    $s_i \leftarrow F(p_i)$ ▷ Evaluate new solution
    **if** non-dominated$(A, s_i)$ **then**
      $A \leftarrow$ update_A$(A, s_i)$ ▷ update archive
      $z^* \leftarrow$ update_Z$(z^*, s_i)$ ▷ update reference point
    **end if**
    **for** $(j, s_j) \in T$ **do** ▷ for each neighbors
      **if** $g(s_i, w_j)$ is better than $g(s_j, w_j)$ **then**
        $P[j] \leftarrow p_i$
        $S[j] \leftarrow s_i$
      **end if**
    **end for**
  **end for**
**until** C

### Decomposition into single-objective sub-problems

The MOEA/D algorithm transforms a multi-objective problem into several single-objective sub-problems. The decomposition process is configured by a set of $\mu$ weight vectors $w_i, i \in \{1, \ldots, \mu\}$ and an aggregation function $g(x, w)$, where $x$ is a solution in the search space $\mathcal{X}$. Each weight vector results in a different single-objective sub-problem to be solved. The population size of the algorithm is equal to the number of sub-problems. Each solution of the population is the current best solution to the corresponding sub-problem. At each generation, MOEA/D iterates over all sub-problems until a stopping criterion is met. At each iteration, the algorithm attempts to improve the current sub-problem. Two parent solutions are first selected to generate a new offspring. The offspring is then modified through a mutation operator. If the newly generated offspring improves the current sub-problem, it replaces the previous best solution of the current sub-problem in the population.

If the number of weights $\mu$ is small, the algorithm spends more time generating new offspring. By contrast, when $\mu$ is large, the number of iterations per sub-problem is reduced if the stopping condition is a fixed number of evaluation. However, the population is larger, hence it may cover the Pareto front more efficiently. Similarly to the number of weight vectors, their distribution alters the efficiency of the algorithm. In the most intuitive approach, weight vectors are distributed uniformly, as illustrated in Figure 1.4. However, the Pareto front may not be uniform. Thus, the algorithm computational efforts devoted to some weight vectors may lead to a loss of function evaluations. For bi-objective optimization problems, a set of uniformly distributed weight vectors can be generated with the following equation:

$$w^i = \left( \frac{1-i}{\mu - 1}, \frac{i}{\mu - 1} \right), \forall i \in \{1, \ldots, \mu\}$$

where the first and the last weight vectors ($w^0 = (1, 0)$ and $w^\mu = (0, 1)$) respectively correspond to the first and the second objective of the problem. For more than two objectives, the generation of weight vectors is based on more advanced procedures such as the simplex lattice design [88]. In this thesis, we will only consider bi-objective problems in our experimentation, although most of our contributions can apply for more objectives.

Single-objective sub-problems are defined by a single scalarizing function using the weight vectors. Several scalarizing functions were proposed in the literature. The two most current ones are the weighted sum and the Chebyshev function.

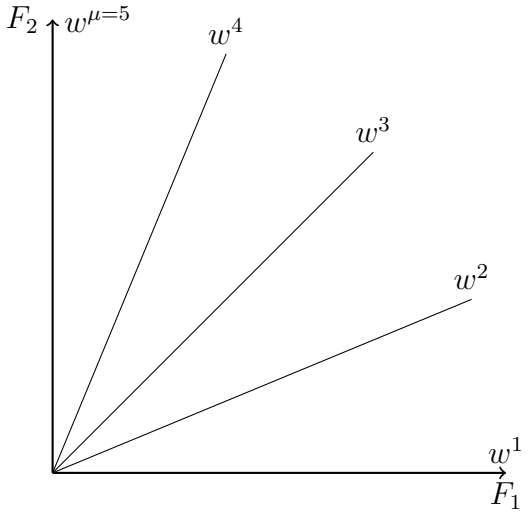


Figure 1.4: Decomposition of a bi-objective problem with 5 weight vectors.

- Given a weight vector $w = (w_1, \ldots, w_M)$, a solution $x \in \mathcal{X}$ and its objective value $F(x) = (F_1(x), \ldots, F_M(x))$, the weighted sum aggregation function is defined as follows:

$$WS(x, w) = \sum_{i=1}^{M} w_i \cdot F_i(x)$$

- The Chebyshev aggregation function compare solutions to an ideal reference point $z^* \in \mathbb{R}^M$. Given a weight vector $w = (w_1, \ldots, w_M)$, a solution $x \in \mathcal{X}$ and its objective value $F(x)$, the Chebyshev aggregation function is defined as follows:

$$g(x, w) = \max_{i \in \{1,\ldots,M\}} w_i \cdot \left| z_i^* - F_i(\mathbf{x}) \right|$$

  The reference point $z^* = (z_1^*, \ldots, z_M^*)$ is expected to be an ideal point. In a black-box scenario, since the maximal value of each objective is unknown, the reference point is set as the best value found so far for each objective:

$$z^* = \max_{x \in \mathcal{X}'} F_i(x), \forall i \in \{1, \ldots, M\}$$

  where $\mathcal{X}'$ is the set of solutions evaluated so far by the algorithm. As the reference point may not be an ideal point once new solutions are generated, it must be updated at each iteration of the algorithm.

### Cooperation between sub-problems

The specificity of the MOEA/D algorithm is that single-objective sub-problems cooperate. Using an integer parameter $V$ to define the size of the neighborhood, all sub-problems are assigned $V$ neighbors. In the bi-objective case, when $V$ is even and given ordered weight vectors $W = \{w_1, \ldots, w_\mu\}$, the neighborhood T of a weight vector $w_i$ is the set of the $V$ closest weight vectors defined as follows:

$$T_i = \left\{ w_j, \forall j \in \{1, ..., \mu\}, j \neq i \text{ such that } |i - j| < \frac{V}{2} \right\}$$

The cooperation affects both the selection and replacement step. For a current sub-problem $w_i$, only its neighborhood $T_i$, a subset of the population, can be selected as a parent solution. In addition, newly generated solutions from the weight vector $w_i$ are compared to solutions from the neighborhood $T_i$ using their respective scalarizing function; hence, possibly improving other weight vectors than the one it was originally assigned to.

### Replacement

In MOEA/D, the newly generated solution can replace one or more solutions. Given the current weight vectors $w_i$, solutions that may be replaced are the previous best solutions of with respect to weight vectors $w_i$ and its neighborhood $T_i$. The solution to be maintained for each weight vector is the one optimizing the corresponding scalarizing function $g(x, w)$.

### MOEA/D variants

Many variants of MOEA/D exist in the literature [62, 63]. In the second and third contribution (Chapter 3 and 4), two variants called MOEA/D-SS and MOEA/D-SC [63] are considered. These are particular variants of a more general framework, called MOEA/D-XY, which modifies the selection (X) and replacement mechanisms (Y) of the standard version of MOEA/D. For both MOEA/D-SS and MOEA/D-SC, the first $S$ stand for *selfish*, meaning that selected parents always include the solution from the current weight vector. The second parent is selected among solutions of the neighbors $T_i$. The second parameter $Y$ affects the replacement strategy. The difference only appears in MOEA/D-SS, as the replacement is the same for MOEA/D and MOEA/D-SC. The replacement in the population is a selfish replacement. The newly generated solution can only replace the previous solution of the current weight vector. By removing the cooperation between sub-problems, the algorithm

favors diversity in its population. A third variant is considered in combination to MOEA/D-SC and MOEA/D-SS, to introduce a maximum number of replacement for a given solution. This third variant is called MOEA/D-NR [62]. It introduces the $nr$ parameter to fix the maximal number of replacements to avoid cases where a good solution outperforms most solutions of the closest sub-problems, resulting in a loss of diversity in the population.

### 1.3.2 Dominance-based Approaches

A second class of approaches is based on the dominance relation. As mentioned earlier, the dominance defines a partial order between solutions. Among dominance-based algorithms, NSGA-II [26] is used in Chapter 5. The main steps of the algorithm are presented in Algorithm 2.

**Algorithm 2** NSGA-II [26]

**Input:** $F = (F_1, ..., F_M) : \mathcal{X} \longrightarrow \mathcal{Y}$: fitness function
$L$: Population size.
$C$: stopping condition.

$P \leftarrow \{$**random solution**$, \forall i \in \{1, ..., L\}\}$ ▷ Initialize population
$S \leftarrow \{F(p), \forall p \in P\}$ ▷ Evaluate the population
$A \leftarrow$ Non-dominated$(S)$ ▷ Initialize archive
**repeat**
▷ generate next Population by selection, recombination and mutation
$P' \leftarrow$ generate()
$S \leftarrow \{F(p), \forall p \in P'\}$ ▷ Evaluate new solutions
ranks $\leftarrow$ non-dominated-sorting$(P', S)$
dists $\leftarrow$ crowding distance$(P', S)$
▷ select solutions for the next generation of population
$P \leftarrow$ sort$(P \cup P',$ ranks, distances$)$
$A \leftarrow$ update Archive$(A, P')$ ▷ update archive
**until** C

The NSGA-II algorithm uses two sorting procedures as a mechanism to select the solution to keep from the union of the previous population $P$ and the offspring $P'$ generated in the current generation. The first sorting procedure is the non-dominated-sorting detailed in Algorithm 3. The function returns the Pareto rank for each solution: non-dominated solutions $P^1$ from the population are assigned a rank of 1. From the remaining solutions of the population $P \backslash P^1$,

non-dominated solutions are assigned a rank of 2 and so on. As such, solutions with the lowest Pareto rank are closer to the Pareto front.

**Algorithm 3** Non-dominated sorting

```
Input: P: population of solution
       S: multi-objective value of the population

R ← 1
while P is not empty do
    for p ∈ P do
        if p is not dominated by P {p} then
            ranks[p] ← R
        end if
    end for
    R ← R + 1
end while
```

The second sorting procedure is the crowding distance detailed in Algorithm 4. As several solutions of the population may have the same Pareto rank, the crowding distance allows ordering solutions with a same rank. In this procedure, solutions are first sorted in ascending order for the value of the $m_j \in \{1, \ldots, M\}$ objective. Solutions on the boundary are set to an infinite value of crowding distance and intermediary solutions to the normalized difference between the previous and the next solutions. As a result, solutions far apart from others are more likely to be selected, thus increasing diversity in the objective space.

### 1.3.3 Indicator-based Approaches

The last category of approaches is based on the use of an *indicator*. A quality-indicator is a set function that allows simpler comparison between solutions or set of solutions. Evolutionary algorithms from this category use a quality-indicator such as the hypervolume [8, 105] or the $R2$ indicators [37, 17]. Most of the time, an indicator transforms the multi-objective value into a single value. Thus, the indicator defines a total order between solutions. Among algorithms of this category, SMS-EMOA [12] is a popular example. From a population $P$, offspring solutions $P'$ are generated to form a larger population $P_2 = P \cup P'$. Then, the algorithm removes solutions with the smallest measure according to the indicator. Indicator-based algorithms are not considered in the thesis. More information can be found in [31, 32, 54].

**Algorithm 4** Crowding distance

**Input:** $P^r$: subset of the population of rank $r$
$S$: multi-objective fitness of the population subset

$n \leftarrow |P^r|$
**for** $i \in \{1, \ldots, n\}$ **do**
  $dists[i] \leftarrow 0$ ▷ initialize distances
**end for**
**for** $m_j \in \{1, \ldots, M\}$ **do** ▷ for each objective
  $sort(P^r, S, m_j)$ ▷ Sort the population by the $m_j$ objective value
  $dists[0] = dists[n] = \infty$
  **for** $i \in \{2, \ldots, n-1\}$ **do**
    $dists[i] = dists[i] + \frac{S_{m_j}[i+1] - S_{m_j}[i-1]}{S_{m_j}[0] - S_{m_j}[n]}$
  **end for**
**end for**

## 1.4 Performance Analysis

The analysis and the comparison of algorithms require a fair comparison and setting, that do not favor one algorithm over the other. As mentioned earlier, evolutionary algorithms rely on a stopping condition. This condition must be the same for all algorithms. It ensures that the execution time or that the resources needed by the algorithms are the same. As for the measure, quality indicators are often used for multi-objective optimization, since they make it easier to compare sets of solutions.

### 1.4.1 Algorithm Budget and Stopping Condition

During the execution of an evolutionary algorithm, the algorithm iterates until a stopping condition is satisfied. To allow simpler comparison of algorithm performance, it is necessary to set comparable stopping conditions to algorithms. As the execution time is the most common issue when solving difficult problems, the execution time is considered as a budget and the algorithm stops once the budget is exhausted. However, physical differences between computer components impact the execution time, making it unreliable to use the exact CPU time. In black-box settings, the evaluation of a solution is considered as the most time-consuming operation. Such a setting allows us to consider the number of calls to the objective function as the budget. Given a budget $B$, once the algorithm has evaluated $B$ solutions, the algorithm stops.

### 1.4.2 Quality Indicators

One of the major difficulties in multi-objective optimization is the absence of total order between solutions and between sets of solutions. As seen previously, it is possible to classify solutions according to the dominance relation. However, this relationship remains a partial order. For two sets, e.g., the sets of solutions found by two algorithms, it is necessary to compare them. For example, in Figure 1.5, the set of red points contains various solutions distributed between the two objectives. In contrast, the set of green points has fewer solutions with a higher concentration of solutions in the center of the objective space. Green solutions have a slightly better aggregated value of the objectives, e.g, the Euclidean distance between red solutions and ideal point is smaller. Using only the dominance relation, all solutions from both the red and green sets are incomparable. It is not reasonable to compare both sets using solely the dominance relation. In fact, each set has its own strength; the red set has a better diversity and the green set is closer to the ideal point.

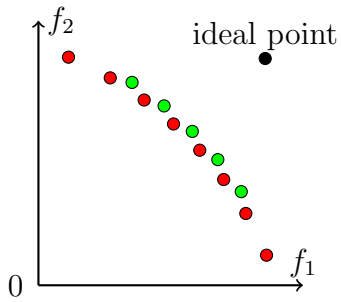


Figure 1.5: Example of two incomparable fronts; Green front has a concentration of solution around the center between objectives and a better convergence to the ideal points, red front has more solutions and an improved diversity in the objective space. All solutions are incomparable.

To compensate for the absence of total order, several quality indicators have been proposed in the literature [109, 106]. Among these, the hypervolume [8, 108] measures the multidimensional portion of the objective space covered by the solution set under consideration. This indicator is a unary indicator, assigning to each approximation set a scalar value. According to the space covered by the solutions of the approximation set, the larger the hypervolume, the better the approximation. In Figure 1.6, the hypervolume of the red and

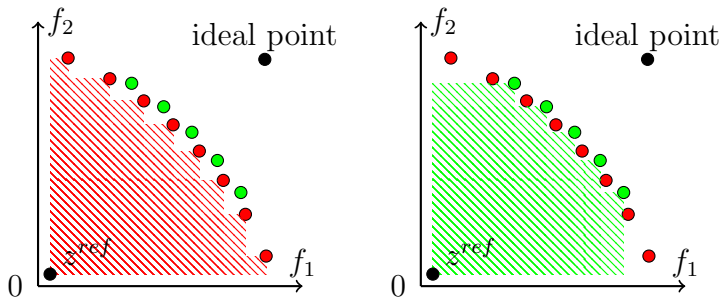


Figure 1.6: Hypervolume of the two previous front from example 1.5. The green respectively red area shows the area covered by the green and red front approximation, respectively.

green set respectively corresponds to the highlighted red and green area. To compute the hypervolume, a reference point $z^{ref}$ is required.

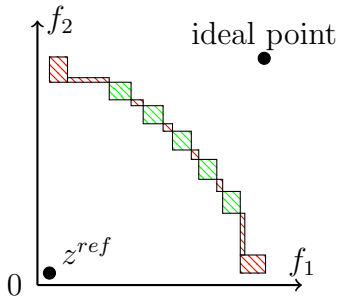


Figure 1.7: Visual representation of the difference of hypervolume between the green and red front. Green areas, respectively red areas, show the portion of space covered by the green approximation but not by the red approximation.

The relative hypervolume deviation is a variant of the hypervolume indicator. This variant allows us to compare two set of solutions directly. It computes the difference of hypervolume, i.e., the difference in the covered space between the two sets (Figure 1.7). It is especially useful to compare the quality of a set in relation to the Pareto set. In the example, both approximations reach a similar hypervolume. However, the hypervolume covered by the green set slightly outperform the hypervolume of the red set.

On top of these considerations, for evolutionary algorithms, the performance varies from one execution to another. To evaluate the performance,

it is hence necessary to execute multiple runs using different random seeds. Statistical tools, such as the mean or the standard deviation, are used on the indicator distribution to draw conclusions on algorithm performance.

## 1.5 Performance Assessment and Benchmarks

For the development of most studies presented in this thesis, one of the first (bottleneck) issues is to fix a collection of optimization problems on which experimental results can be derived.

### 1.5.1 Importance of Benchmark

There are a number of studies [10] that highlight the advantages of the development of benchmarks and their use. This is especially important for the visualization and the comparison between publications. Some benchmarks are made of a set of known problems, such as the black-box optimization benchmarking test-suites (BBOB) [38, 16] for continuous optimization. Others are *framework-type* benchmarks, such as the NK-landscapes [9] or the W-Model [97]. In these case, the benchmark is a problem generator. It uses several parameters to generate problem instances with specific characteristics. In this thesis, the benchmark needs to have the following requirement:

1. Artificial combinatorial optimization problem: as per choice this study focuses on landscape analysis for multi-objective combinatorial optimization, this is a mandatory aspect for benchmarking the approaches under consideration.

2. Scalability of the problem: the dimension and the number of objectives must be tunable to generate a diversified set of problem instances.

3. Well-known problem characteristics and landscapes: as this thesis relies on the design and the analysis of multi-objective landscape features, insights on the problem structure simplify the understanding of the proposed methods.

Following these requirements, we selected the $\rho$MNK-landscapes as a multi-objective problem generator. The benchmark is defined in the next section.

### 1.5.2 $\rho$MNK-Landscapes

$\rho$MNK-landscapes constitute a multi-objective combinatorial optimization benchmark. It is actually a framework for generating multi-objective problems with different characteristics. Generally speaking, NK-landscapes [46] are challenging single-objective combinatorial optimization problems. At first, they were proposed to investigate the relationship between the epistasis and the landscape. The epistasis is a biological notion that characterizes the interactions between genes (i.e., variables for combinatorial problems). NK-landscapes are configured by two integers: $n$ the number of binary variables and $K$ the number of interaction between variables.

Given a number $n$ of binary variables, an integer parameter $K < n$, a set of $K$ interactions $\Pi(x_j) \subseteq \{x_1, \ldots, x_n\} \setminus \{x_j\}$ for each variable $x_j$, $j \in \{1, \ldots, n\}$ with $|\Pi(x_j)| = K$, and a real-valued sub-function $f_j$ defining the contribution of each combination of values of $x_j$ and $\Pi(x_j)$, the (single-)objective function $F$ underling a NK-landscape (to be maximized), is defined as follows:

$$F(\mathbf{x}) = \frac{1}{n} \cdot \sum_{j=1}^{n} f_j(x_j, \Pi(x_j)). \tag{1.2}$$

It is important to notice that NK-landscapes are basically a linear combination of non-linear sub-functions (contribution) of $K + 1$ variables.

MNK-landscapes [2] are an extension of the NK-landscapes for multi-objective optimization. A MNK-landscape is defined as a vector function mapping every binary string $\mathbf{x} \in \{0, 1\}^n$ into $M$ real numbers $\mathbf{F} = (F_1, F_2, \ldots, F_M) : \{0, 1\}^n \to \mathbb{R}^M$, where $M$ is the number of objectives, $n$ is the number of variables, $K$ is the number of interactions per variable, and $F_i$ is the $i$-th objective function defined by, $i \in \{1, \ldots, M\}$:

$$F_i(\mathbf{x}) = \frac{1}{n} \cdot \sum_{j=1}^{n} f_j^i(x_j, \Pi(x_j)). \tag{1.3}$$

As in the single-objective case, $\Pi(x_j)$ denotes the set of $k$ variables interacting with $x_j$ within each of the sub-functions $f_j^i : \{0, 1\}^{K+1} \to \mathbb{R}$.

An extension to MNK-landscapes adds correlation between objectives [94]. In $\rho$MNK-landscapes, fitness components $f_j^i$ are not defined independently. For all solutions $\mathbf{x}$, its associated value $f_j^i(x_j, \Pi(x_j))$ follows a multivariate uniform distribution of dimension $K$, defined by a correlation matrix $C_\rho$. This is a symmetric positive-definite used to define random variables with specified level of correlation [40]. When $\rho = 0$, no correlation is introduced, hence

$\rho$MNK-landscapes are similar to MNK-landscapes. For a positive value: $0 < \rho < 1$, objective values are positively correlated, leading to solutions with good objective value for each objective. Inversely, when $\rho$ is negative: $-1 < \rho < 0$, objectives are conflicting and improving an objective seemingly reduces the objective value for other objectives. It is important to remark that, in the standard definition of $\rho$MNK-landscapes and in order to control the objective correlation, the size and the variables of $\Pi(x_j)$ do not depend on the objective function, i.e., every variable has exactly the same set of interactions for each objective, in contrast to MNK-landscapes.

## 1.6 Conclusion

In this chapter, we introduced various concepts related to multi-objective optimization. These concepts include the definition of a multi-objective optimization problem and their Pareto optimal solutions, defined by the dominance relation. We introduced the main resolution approaches, and we thoroughly explained how evolutionary algorithms operate to solve an optimization problem. MOEA/D and NSGA-II were presented in details, as both state-of-the-art multi-objective evolutionary algorithms are used in the following chapters. As this study focuses on black-box optimization, we emphasized the importance of quality indicators and algorithm budget to analyze algorithm performance. Finally, we discussed the importance of benchmark, and we introduced $\rho$MNK-landscapes, the multi-objective combinatorial benchmark used for experimental studies in the following chapters.

In the next chapter, we present the first contribution of this thesis. Inspired by the decomposition paradigm from MOEA/D, we propose to decompose the multi-objective optimization problem into single-objective sub-problems to analyze the landscape. The presented approach is empirically tested and analyzed on $\rho$MNK-landscapes.

# Chapter 2

# Decomposition-based Landscape Analysis

In this chapter, we present the first contribution of this thesis, which deals with the design of a decomposition-based methodology for multi-objective landscape analysis. Firstly, the necessary background on landscape analysis is provided. Secondly, the new methodology for computing multi-objective features is introduced. The proposed approach is inspired by the concept of decomposition as introduced in the previous section within the state-of-the-art MOEA/D algorithm. Thirdly, a first experimental analysis using $\rho$MNK-landscapes is conducted to highlight the relevancy of the proposed approach.

The contribution presented in this chapter has been published and presented at the European Conference on Evolutionary Computation in Combinatorial Optimization (EvoCOP 2021): "Decomposition-based landscape analysis for multi-objective optimization" and it was *nominated* for the best paper award [21].

## 2.1 Fitness Landscape Analysis

The concept of adaptive landscape (or fitness landscape) was introduced by S. Wright in the field of evolutionary biology in the early thirties [101]. It consists in representing living organisms using an abstract space with a neighborhood relationship. This concept has also been established in other fields of science such as molecular biology, statistical physics [36, 71, 46], or combinatorial optimization to interpret complex dynamic systems. In the field of combinatorial optimization, the interest of this metaphor is to define an abstract structure of the search space and to link it to the dynamics of search algorithms, in particular by extracting useful information about the difficulty for an algorithm to optimize a given problem.

### 2.1.1 Definition and Background

With respect to single-objective optimization problems, a fitness landscape can be defined as a triplet $(\mathcal{X}, \mathcal{N}, F)$, where:

- $\mathcal{X}$ is the solution space (also called search space).
- $\mathcal{N} : \mathcal{X} \longrightarrow 2^{\mathcal{X}}$ is a neighborhood relation which maps for each solution $x \in \mathcal{X}$ a set of neighbors $\mathcal{N}(x)$.
- $F : \mathcal{X} \longrightarrow \mathbb{R}$ the fitness function that associate a scalar value $F(x)$ to each solution $x \in \mathcal{X}$.

The search space $\mathcal{X}$ and the fitness function $F$ are in general defined very naturally with respect to the optimization problem at hand. The neighborhood relation can however be defined in several manners, as it is the case when designing a local search algorithm. Depending on the problem at hand, different neighborhood relations can be considered; hence, leading to different landscapes.

In combinatorial optimization, the neighborhood function depends on how solutions are represented. For binary optimization problems, a solution $x \in \mathcal{X}$ is a binary vector $x = \{0, 1\}^n$. The most common neighborhood relation is the bit flip and, more generally, the distance between two solutions is the Hamming Distance $\mathcal{D}$. The Hamming Distance between two solutions $x, y \in \mathcal{X}^2$ is the number of bits that differs in $x$ and $y$. When the Hamming Distance between solutions $x$ and $y$ is $D(x, y) = 1$, $x$ and $y$ are said to be neighbors. In this case, the neighborhood relation is symmetric, i.e., if $y$ is in $\mathcal{N}(x)$ then $x$ is in $\mathcal{N}(y)$.

It is important to notice that different neighborhood relations result in different landscapes. Fitness landscapes are not necessarily three dimensional landscapes. In the case of combinatorial problems using the hamming distance, each solution is a binary vector with $n$ neighboring solutions and therefore the landscape has $n + 1$ dimensions.

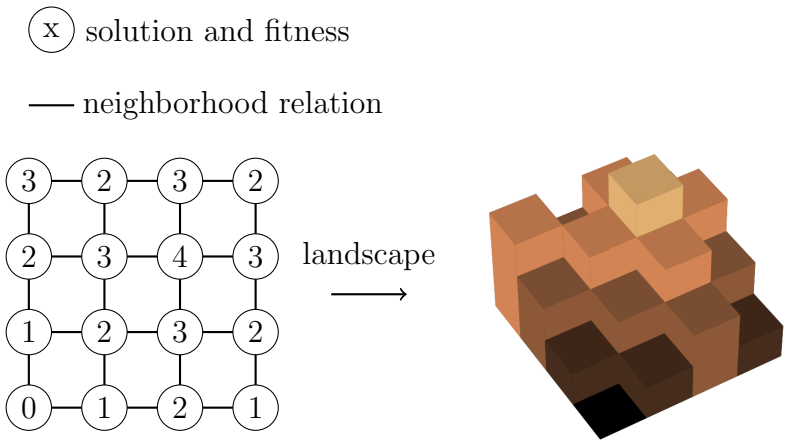


Figure 2.1: Schematic transformation from the search space and fitness values to a landscape.

In a three-dimensional landscape, such as in Figure 2.1, each of the solutions is a point which can be sorted by their proximity. Due to the number of dimension, it is not possible to easily visualize the landscape of combinatorial problems. In the example, the resulting $\Gamma = (S, V, F)$ landscape graph is a set of connected vertices (solutions), edges indicate the neighborhood relation between two solutions and the value of all vertices is the fitness value.

### 2.1.2 Landscapes and Problem Structures

Fitness landscape analysis is particularly interesting for observing and characterizing the structural properties of a given optimization problem instance. Besides, the intrinsic characteristic of a problem are known to impact the performance of the considered optimization process. In particular, the "no free lunch" theorem [99, 100] states that all optimization algorithms have equivalent average performance over the set of fitness functions $\mathcal{F} : \mathcal{X} \longrightarrow \mathcal{Y}$, where $\mathcal{X}$ is the solution space, $\mathcal{Y}$ the fitness space and both $\mathcal{X}$ and $\mathcal{Y}$ are finite sets.

As a result, if there is an optimization algorithm $A_1$ that outperforms the algorithm $A_2$ on a subset of $\mathcal{F}$ functions, then there is another subset of $\mathcal{F}$ where $A_1$ performs worse than $A_2$. The consequences of this theorem have been thoroughly discussed [74, 83, 42]. However, this theorem reinforces the idea that the development of an algorithm requires knowing the characteristics of the problem that is to be solved.

**Local optima:** Given an optimization problem $\mathcal{P} : \mathcal{X} \rightarrow \mathbb{R}$, a **local optima** is a solution $x \in \mathcal{X}$ for which there exist no better solutions in its neighborhood $\mathcal{N}(x)$. Therefore, the optimal solution of the problem is a local optima. However, a local optima is not necessarily an optimal solution. To make a distinction, the optimal solution is called **global optima**. In multi-objective optimization, we make a distinction between **Pareto Local Optima** (PLO) and **single-objective local optima** (SLO) which are local optima with respect to only one objective. As illustrated in Figure 2.2, the landscape can have more than one local optima. In the figure, each peak is a local optimum. Evolutionary algorithms, especially basic search algorithms such as Hill

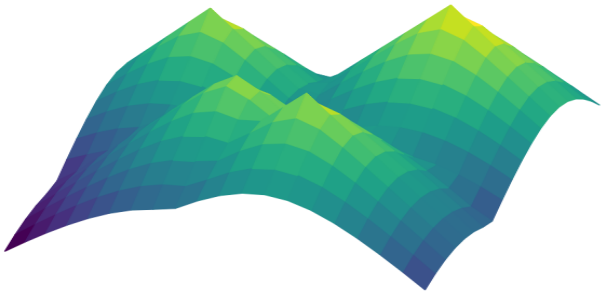

Figure 2.2: 3-dimensional representation of Local optimas.

Climbers [44], evolve a solution by selecting the best solution in the neighborhood. Thus, once the solution is a local optima, algorithms get stuck and cannot improve. In recent years, although a number of approaches have been proposed to tackle this difficulty [39], landscapes with many local optima solutions remain hard to solve in general.

**Smooth and unimodal landscapes:** A landscape is said to be unimodal if the only local optima is the global optima. An example of a unimodal landscape is illustrated in Figure 2.3. The landscape is smooth if for all solutions from the search space, there is at least one neighbor which improves the fitness, except for the global optima. Due to this characteristic, smooth

and unimodal landscapes are considered to be easy to solve. It is possible to determine the direction to evolve a solution towards the optimum. Therefore, given any solution of the search space, there is a path leading to the global optima by performing successive bit-flip operations.

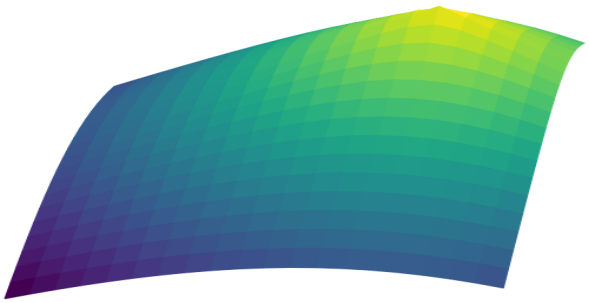

Figure 2.3: 3-dimensional representation of a smooth landscape.

**Rugged landscapes:** One of the most important landscape characteristic is about its ruggedness [41]. A landscape is said to be rugged when its local structure is rough, as for example in Figure 2.4. Rugged landscapes are the opposite of smooth landscapes. These landscapes are irregular, and neighbors can have major gaps in their fitness. For Hill Climber algorithms, neighbors improving the current solution are not necessarily the best direction to follow, as the paths to arrive to the global optimal solutions are not made up solely of improving solutions.

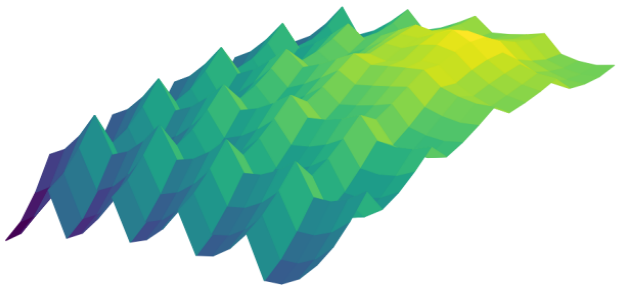

Figure 2.4: 3-dimensional representation of a rugged landscape.

**Neutral landscapes:** Another classical landscape property concerns neutrality [92], as illustrated in Figure 2.5. Neutral landscapes contain solutions

whose fitness values are the same as their neighbors. Such landscapes contain plateaus with a number of neighboring solutions of equal fitness. Such landscapes are also difficult to search for evolutionary algorithms, as there is no relevant information to decide on the direction to take.

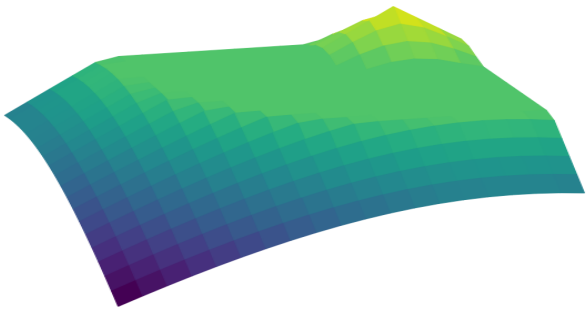

Figure 2.5: 3-dimensional representation of a neutral landscape.

### 2.1.3 Landscape Features

As mentioned previously, landscapes with more than three dimensions are difficult to interpret. Besides, the commonly used metaphor, with peaks and valleys, can be misleading. Generally speaking, the definition of a fitness landscape allows one to compute numerical features that provides useful information about the optimization problem at hand. Such information might in fact not be available or easy to derive from the original definition of the problem. More specifically, landscape features are in general defined by extracting some statistical information describing the so-defined landscape. Such features can be typically used for different purposes, e.g., differentiate between different instances of the same problem, differentiate between different problem classes, predict the performance of a given algorithm on a given instance, choose the best performing algorithm among a given portfolio for a given instance, etc.

**Global features, local features and sampling strategies:** In order to compute the features of a given landscape, the search space needs to be evaluated. When features are computed using the complete landscape as data, then the features are termed **global**. Such metrics provide relatively precise information about the landscape structure, but they heavily depend on the full enumeration of the search space. While this can help to improve our understanding of problem structures, it is however not a reasonable approach for landscapes with high dimensions. For problem with large dimensions, features

**Algorithm 5** Random walk [4]

**Input:** $l \in \mathbf{N}$: length of the walk.
$F : \mathcal{X} \longrightarrow \mathcal{Y}$: fitness function.

$x \leftarrow$**random solution** ▷ set the initial solution
$S \leftarrow \{(x, F(x))\}$ ▷ evaluate the solution
$c \leftarrow 0$
**repeat**
  **for** $x_i \in \mathcal{N}(x)$ **do** ▷ for each neighbors of $x$
    $S$.append($\{(x_i, F(x_i))\}$) ▷ evaluate the solution
  **end for**
  $c+ = 1$
  $x \leftarrow$ *Uniform Random*($\mathcal{N}(x)$) ▷ select the next solution at random
**until** $c = l$
**return** $S$

are typically computed using a sample of solutions. By using randomness in the sampling strategy, and if the sample is large enough, the computed features are expected to be representative of the landscape. Features computed with a sample are termed **local features**. Several local features have been proposed in the last decades. In fact, the so-called Exploratory landscape analysis (ELA) [64, 48, 51] has emerged as a state-of-the-art methodology for single-objective continuous optimization. The approach combine the extraction of local landscape feature with the development of automated recommendation systems integrating the so-considered features on the basis of statistical or machine learning models in order to solve different challenging tasks.

There exist many sampling strategies, among them random and adaptive walks [4, 47] are used in this work. Random and adaptive walks are iterated process which move repeatedly in the landscape. The main steps of random and adaptive walks are detailed in Algorithm 5 and Algorithm 6 respectively. Both walks start with a random solution $s^0$ and select the next solution $s^{i+1}$ in its neighborhood $\mathcal{N}(s^i)$. In a random walk, there is no particular criterion to pick the next neighboring solution. Starting with a first (randomly chosen) solution $x^{t=0}$, at each iteration $t \in \{0, \cdots, \ell - 1\}$, the next solution $x^{t+1}$ is chosen uniformly at random among the neighbors $\mathcal{N}(x^t)$ of the solution $x^t$. The walk length $\ell$ is a user-defined parameter. The longer the walk, the larger the sample and the better the information.

In an adaptive walk, the next solution is a solution which improves the fitness among the neighbors in $\mathcal{N}(x^t)$. Unlike a random walk, the length $\ell$ of an

**Algorithm 6** Single-objective adaptive walk [47]

**Input:** $F : \mathcal{X} \longrightarrow \mathcal{Z}$: fitness function.

$x' \leftarrow$**random solution** ▷ set the initial solution
$S \leftarrow \{(x, F(x))\}$ ▷ evaluate the solution
**repeat**
  $x \leftarrow x'$
  **for** $x_i \in \mathcal{N}(x)$ **do** ▷ for each neighbors of $x$
    $S$.append($\{(x_i, F(x_i))\}$) ▷ evaluate the solution
  **end for**
  $x' \leftarrow \max(\{F(x_i) | \forall x_i \in \mathcal{N}(x)\})$
**until** $f(x) > f(x')$ ▷ (for maximisation)
**return** $S$

adaptive walk is determined by the number of steps required to fall into a local optima $x^\ell$. Multiple adaptive walks are performed to get reliable information. In the case of a multi-objective problem, a multi-objective adaptive walk selects a dominating neighbor and stops when all neighbors are dominated or incomparable. We call such walk *dominance walk*.

### 2.1.4 Multi-objective Landscapes Features

The landscape of a multi-objective problem is similarly defined as a triplet $(\mathcal{X}, \mathcal{N}, F)$. However, the fitness function is actually multi-objective $F : \mathcal{X} \rightarrow \mathbb{R}^M$. Thus, numerical features for single-objective problems are not compatible with such a landscape. Two approaches have been proposed to overcome this difficulty, and multi-objective features were developed [55]. Existing features can be classified into two categories:

- Features based on dominance. This category extracts statistics on the basis of the dominance relation when comparing a solution with its neighbors. For example, one may compute some statistic about the number of incomparable or dominating neighbors.

- Features based on indicators. Roughly speaking, the idea of these features is to compute for each solution a numerical value using a quality indicator as discussed earlier in Chapter 1, e.g., the hypervolume contribution with respect to a given set. Hence, one may consider such values to extract some statistics about the landscape instead of using the original multi-dimensional objective vector.

Besides these two main classes, one may find other features considered for multi-objective optimization. For example, Kerschke et al. [50] proposes to compute single-objective features independently for each objective. Such an approach does not require new features dedicated to multi-objective optimization, and it shows that the multi-objective landscapes generated by the combination of single-objective landscapes are strongly related to the single-objective landscapes. However, this method is limited. For example, it is not clear that the union of the different single-objective landscapes can fairly represent the overall characteristic of the implied multi-objective landscape. In particular, it may happen that a single-objective landscape appears to be easy to search for a given algorithm; however, when combined with other objectives, the implied multi-objective landscape may have a different degree of difficulty. Additionally, in such an approach, a set of features $L$ is computed for each objective, resulting in $M * |L|$ features, and the number of features depends on the number of objectives. Therefore, the numerical characterization of the landscape does not make it possible to compare the landscape for problems with different dimensions in the objective space.

## 2.2 Decomposition-based Landscape Analysis

In this thesis, we get inspiration from the concept of decomposition used in a number of multi-objective search algorithms [91] in order to design new high-level multi-objective features. In fact, the concept of decomposition allows designing algorithms that search for good-performing solutions in multiple regions of the Pareto front. This is by decomposing the original multi-objective problem into a number of scalarized *single-objective sub-problems*. Each sub-problem is obtained by a different parameterization of the same underlying scalarizing function. This is typically what the state-of-the-art MOEA/D algorithm [104] performs, while introducing a cooperation mechanism among sub-problem solving. In particular, this offers much flexibility for integrating existing single-objective search operators and solvers, which is actually one of the main reasons for the success of decomposition-based multi-objective evolutionary algorithms. Let us however recall that the main goal of this chapter is *not* to design a new multi-objective algorithm, but to design new multi-objective features that can later feed the design of a general-purpose landscape-aware solving approach.

Therefore, we propose to rely on the simple observation that each sub-problem defined by decomposition also implies a single-objective landscape that we can attempt to analyze and characterize. In other words, by study-

ing the single-objective landscape implied by the sub-problems, we should be able to extract some knowledge about the original multi-objective problem. More precisely, the methodology that we adopt in the reminder of this contribution consists in: (i) defining a number of single-objective landscapes using decomposition, (ii) extracting features for each sub-problem landscape, and (iii) aggregating those scalar features into new multi-objective features. These steps are detailed below.

### 2.2.1 First Step: Decomposition using Weight Vectors

Firstly, we define $\mu$ scalarized single-objective sub-problems, where both the scalarizing function and the $\mu$ value are user-defined parameters. There exists several scalarizing functions, among them Weighted Sum and Chebyshev are the most common as commented in Chapter 1. Let us recall that weighted sum considers a convex combination of all objectives. It is well-known that such a convex combination does not allow achieving all Pareto optimal solutions. In contrast, it is also well known that any Pareto optimal solution can be achieved by solving a well parameterized (with respect to the weight vectors) Chebyshev function. Hence, in the rest of this thesis, we decide to use the Chebyshev function. Notice that using a different aggregation function will not lead to the same features values.

As all following studies suppose that we are in a black-box setting, we do not know how solutions are distributed in the objective space. The difference of maximal and minimal objective values may be different between the objectives. In order to use weight vectors uniformly distributed in the objective space, we consider the normalization of objective values as proposed in the original MOEA/D article [104]:

$$\hat{f}_i(x) = \frac{f_i(x) - \min_{\forall x \in \mathcal{X}} f_i(x)}{\max_{\forall x \in \mathcal{X}'} f_i(x) - \min_{\forall x \in \mathcal{X}'} f_i(x)} \tag{2.1}$$

where $\mathcal{X}'$ is the set of explored solutions. It is worth noticing that we do not make any assumption about the original (black-box) multi-objective problem, so that we have no information about what region every sub-problem is actually mapping. Hence, the value of $\mu$ as well as the procedure to generate weight vectors can be a critical issue. This is studied in more details later when reporting our empirical results.

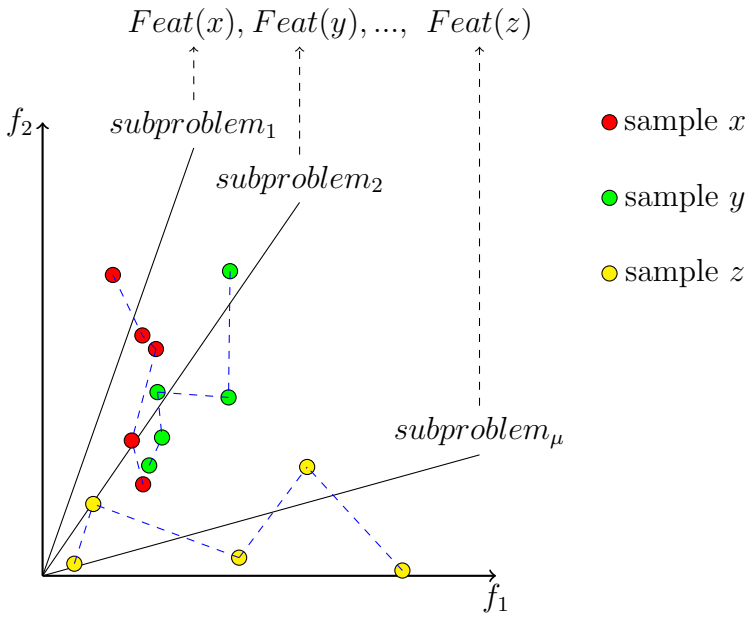


Figure 2.6: Scalar features computation for each sub-problems.

### 2.2.2 Second Step: Scalar Features Computation

Next, we define a landscape for every single-objective sub-problem $j \in \{1, \ldots, \mu\}$, for which we compute a number of underlying high-level scalar features. The scalar features are single-objective features from the literature. In order to differentiate them as they are computed on sub-problems, we call them scalar features in the rest of the thesis. Following the standard literature on single-objective landscape analysis [78], the landscape of sub-problem $j$ can be defined as a triplet $(\mathcal{X}, \mathcal{N}, \mathsf{g}(\cdot, w_j))$, such that $\mathcal{N} : \mathcal{X} \longrightarrow 2^{\mathcal{X}}$ is a neighborhood relation defined among solutions for the considered problem; e.g., 1-bit-flips for binary strings, or swaps for permutations.

The considered sub-problem features are based on sampling the so-defined landscape to compute some statistics. As explained in Section 2.1.3, we follow the standard literature [78, 46, 96], and we consider two simple sampling strategies, namely random walks (rws) and single-objective adaptive walks (aws). More specifically, in this chapter, the sampling strategy is independent for all sub-problems; a distinct walk is computed for each sub-problem as defined by it parameterized Chebyshev function, as illustrated in Figure 2.6. For

each weight vector $w_u$ with $u$ in $\{1, \ldots, \mu\}$, its respective aggregation function is generated, and is used in the walk to evaluate the single-objective value of solutions. When a random walk is used, the cost of the sampling is of $\ell * n$ where $n$ is the problem dimension and $\ell$ is the length of the walk. As a random walk is computed for each weight vector, the cost of the sample is $\mu * \ell * n$. In the case where an adaptive walk is used for each problem, the length of the walk cannot be configured. The number of evaluated solutions is unknown. Notice that the reference point $z^*$ required for computing the scalar fitness values is updated on the basis of the best objective values seen so far during the walk. In the following of this document, performing independent walks for each sub-problems is alternatively called *scalar walks*.

### 2.2.3 Third Step: Scalar Features Aggregation

In the third step, as illustrated in Figure 2.7, scalar features computed on each sub-problem are merged into a multi-objective feature. One of the difficulty

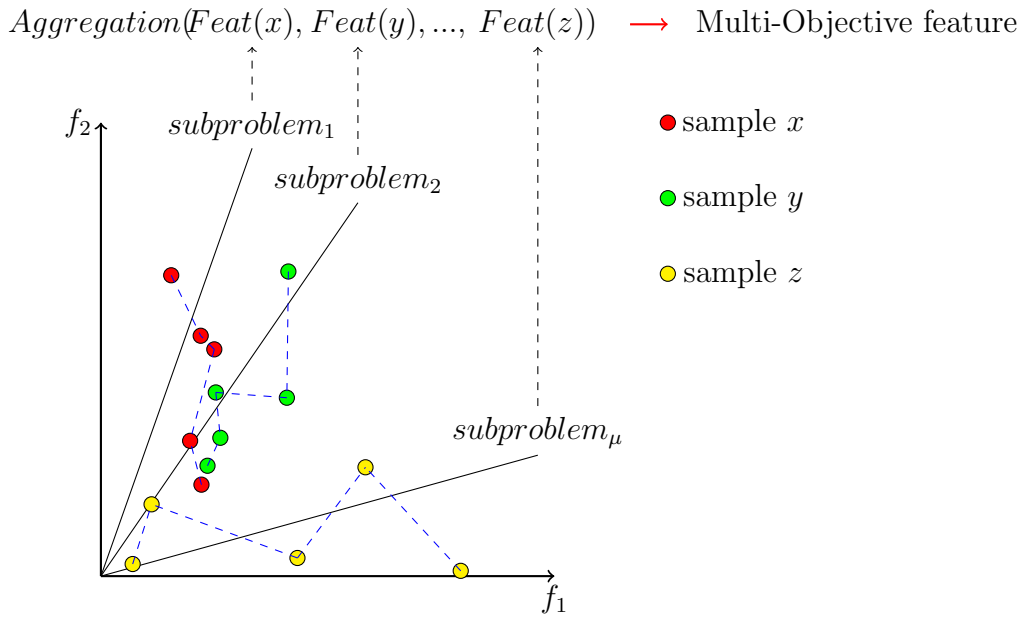


Figure 2.7: Aggregation of the scalar single-objective features into a new multi-objective feature.

in using landscape features is the number of numeral values generated. As

the same feature is computed on $\mu$ sub-problems, it is necessary to aggregate features into a single numeric value. For $\mu > 1$, we need to aggregate these $\mu$-dimensional scalar features into 1-dimensional multi-objective features. For this purpose, we use two standard statistics, namely the mean (avg) and the standard deviation (sd). In addition, we use a polynomial regression in order to fit each scalar feature as a function of the weight vector $w_j$ of sub-problem $j$. The coefficient of the polynomial model are then used as additional aggregated features. We consider a second order polynomial regression and propose to use the first (p1) and the second (p2) coefficients.

### 2.2.4 List of Features

Given the previous methodology, we are then able to compute a set of multi-objective features, as summarized in Table 5.2 and listed exhaustively in Appendix 6.2. More precisely, given a sub-problem $j \in \{1, \ldots, \mu\}$ and a walk $(x_0, x_1, \ldots, x_\ell)$, we consider the following five classes of single-objective features:

- **Fitness value (fv_*).** In the first class, we compute some statistics informing about the distribution of fitness values observed along the walk. Let us recall, that the fitness values are not the original multi-objective fitness but the fitness of the single-objective sub-problem.
- **Fitness difference (fd_*).** In the second class, we compute the average fitness difference with the neighboring solutions for every $x_i$, $i \in \{1, \ldots, \ell\}$: $\frac{1}{|\mathcal{N}(x_i)|} \sum_{x \in \mathcal{N}(x_i)} (\mathsf{g}(x_i, w_j) - \mathsf{g}(x, w_j))$.
- **Improving neighbors (in_*).** In the third class, we compute the number of improving neighbors for each solution $x_i$, $i \in \{0, \ldots, \ell\}$. It is worth noticing that the second and third classes of features require evaluating the fitness value of neighbors from all solutions from the walk.
- **Length of the adaptive walk (law_*).** The fourth class only contains features extracted from the adaptive walk. In particular, we consider the length of the adaptive walk as a feature to characterize the sub-problem landscape. This length was shown to provide an estimation of the number of local optima in single-objective landscape analysis [46].

**Features normalization:** For some features, the possible range of the feature value is related to the problem dimension $n$. For example, the number of improving neighbors is at most $n$. Hence, given two problem instances with

dimension $n_1$ and $n_2$, $n_1 \neq n_2$, the number of improving neighbors can be the same while the local structure is drastically different. To increase the compatibility of the features on different problem instances, we consider the following normalization methods:

- Fitness value: the fitness value of a solution is already normalized according to equation 2.1.
- Fitness difference: instead of the exact value of fitness difference between two solutions $x, x'$, we consider the proportional fitness difference $fd(x, x') = \frac{|F(x)-F(x')|}{F(x)}$.
- Improving neighbors: we consider the number of improving neighbors proportional to the dimension $in(x) = \frac{1}{n} \times \sum_{x' \in \mathcal{N}(x)} 1$ if $F(x') > F(x)$.

**Statistics on the features:** For each feature, we consider several statistics tools. The functions are used to aggregate the intermediary feature values per solution. All functions result in a different (single-objective) feature. The functions are characterized by $s$ in Table 5.2 and are:

- the min and the max (min, max)
- The mean (avg)
- The standard deviation (sd)
- the first auto-correlation coefficient [46, 96] (r1) used in the landscape analysis literature for quantifying the ruggedness of the landscape. Denoting by $\bar{\mathsf{g}}(\cdot, w_j)$ the average fitness value of solutions in the walk, the first auto-correlation coefficient is defined as follows:

$$\mathsf{r1} = \frac{\sum_{t=0}^{\ell-1} (\mathsf{g}(x_t, w_j) - \bar{\mathsf{g}}(\cdot, w_j)) \cdot (\mathsf{g}(x_{t+1}, w_j) - \bar{\mathsf{g}}(\cdot, w_j))}{\left(\sum_{t=0}^{\ell-1} \mathsf{g}(x_t, w_j) - \bar{\mathsf{g}}(\cdot, w_j)\right)^2}$$

- the kurtosis [110] (kr): it measures the tailed-ness of a distribution and is defined as $\mathsf{kr}(X) = \left[\left(\frac{X-\mu}{\sigma}\right)^4\right]$
- the skewness [110] (sk): it measures the asymmetries in the distribution and is defined as $\mathsf{sk}(X) = \left[\left(\frac{X-\mu}{\sigma}\right)^3\right]$
- the first coefficient a polynomial regression (c1) of fitness values.

As we have a total of 192 features, only a subset of them is considered in our experiments.

Table 2.1: A summary of the proposed decomposition-based landscape features.

| description | sub-problem features | MO features |
|---|---|---|
| Fitness values | fv_s | fv_$s$_$r$ |
| Fitness difference | fd_$s$ | fd_$s$_$r$ |
| Improving neighbors | in_$s$ | in_$s$_$r$ |
| Walk length | law_$s$ (only for adaptive walk) | law_$s$_$r$ |

$s \in \{\mathsf{avg}, \mathsf{sd}, \mathsf{c1}, \mathsf{r1}, \mathsf{min}, \mathsf{max}, \mathsf{kr}, \mathsf{sk}\}$ $r \in \{\mathsf{avg}, \mathsf{sd}, \mathsf{p1}, \mathsf{p2}\}$

## 2.3 Experimental Results

As a first step, we analyze the efficiency of the proposed features in capturing the characteristics of multi-objective optimization problems, regardless of any particular evolutionary algorithm. Therefore, we conduct a preliminary exploratory analysis in order to highlight the association between the designed features and the properties of well-established benchmark landscapes.

As introduced in Chapter 1 and following previous works [55], we consider $\rho$MNK-landscapes [95] as a problem-independent model for constructing multi-objective multimodal landscapes with objective correlation. Let us recall that the benchmark has four parameters: the problem instance dimension $n$, the correlation factor $\rho$, the number of objective $M$ and the number of epistatic interactions between variables $K$. Negative and positive values of $\rho$ mean that objectives are conflicting or non-conflicting respectively. Values of $K$ from 0 to $(\mathtt{n}-1)$ tune the landscape from smooth to rugged. In a black-box setting, it is important to recall that parameters $K$ and $\rho$ are *unknown*.

We focus on bi-objective landscapes, i.e., $M = 2$. We used a Latin hypercube sampling [43] to generate a set of 1 000 balanced instances spanning parameters ranges: $n \in \{50, 51, \ldots, 200\}$, $K \in \{0, 1, 2, \ldots, 8\}$ and $\rho \in ]-1, 1]$. The random walk length is set to $\ell = 1\,000$ across all problem sizes. One random walk and one single-objective adaptive walk are computed for each sub-problem. The number of sub-problems is set to $\mu = 20$ and weight vectors are distributed uniformly as explained in Chapter 1. The neighborhood relation is the standard 1-bit-flip.

### 2.3.1 Distribution of Scalar Features

For our analysis, we first conduct an exploratory analysis to better visualize and understand the proposed features. Next, we construct a regression model to study the accuracy of features in grasping the global properties of $\rho$MNK-landscapes, and we analyze the correlation among features.

In Figures 2.8 and 2.9, we report the values of some single-objective features as a function of sub-problems (horizontal axis), computed respectively with adaptive walks and random walks. The blue curves correspond to $\rho$MNK-landscapes with $K = 0$, the green ones to $K = 2$ and the red ones to $K = 4$. The color scales from red to orange, green to cyan, and blue to purple, respectively, and correspond to the objective correlation parameter $\rho$ varying from 1 to $-1$; i.e., from highly correlated to highly conflicting objectives. For example, the standard deviation of fitness values from a random walk (fv_rws_sd, bottom left), the average fitness difference from a random walk (fd_rws_avg, second line, second column) and the standard deviation of improving neighbors (in_rdw_sd, second line, third column) gives a clear differentiation between landscapes with different K-values. The smaller the benchmark parameter K, the smaller the standard deviation of fitness values and the average fitness difference. Similarly, the standard deviation of fitness values from an adaptive walk (vf_aws_sd, top left) and the length of an adaptive walk (law_aws_avg, top right) seems to be clearly associated with parameter $\rho$. The flatter the curve rendering the evolution of these two features as a function of weight vectors, the higher the objective correlation $\rho$.

From our visual inspection, we can conclude that the landscape features are representative of the different *global* benchmark parameters, which are *unknown* in a black-box optimization scenario. However, this first analysis considers the scalar features and not the aggregated 1-dimensional multi-objective features, which are discussed next.

### 2.3.2 Correlation Analysis between $\rho$MNK-landscapes Parameters and Decomposition-based Features

We now report the Spearman correlation matrix among a subset of features and parameters $n$, $\rho$ and $K$ for all problem instances in our data set and a hierarchical clustering of the features in Figure 2.10. The 40 features selected are either the most correlated or most anti-correlated with any of the previous parameters: selected features scored a correlation of at least 0.7 for correlated features or below $-0.7$ for anti-correlated feature. We used decomposition-based features set with $\mu = 50$ and using the scalar sampling previously in-

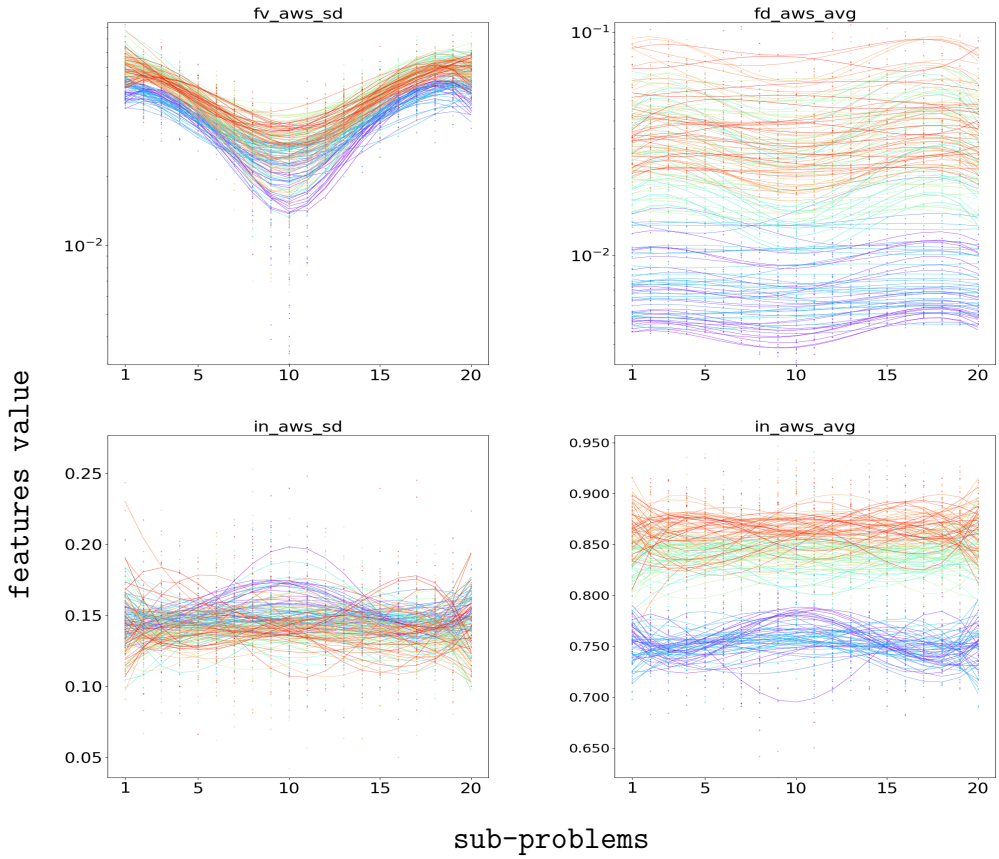


Figure 2.8: Feature values of $\rho$MNK-landscapes decomposed into 20 sub-problems using an adaptive walk($n = 25$, each color correspond to a particular configuration of $\rho$ and $K$).

troduced. This highlight similarities between features and $\rho$MNK-landscapes parameters. It is worth mentioning that among the subset of most correlated or anti-correlated features, each subset of decomposition-based features are represented (*fv, fd, in, law*).

### Cluster associated with Epistasis

The largest cluster (top left of Figure 2.10) with more than 20 features is strongly anti-correlated with $\rho$ but is also partially correlated with $K$. Most features from this cluster belong to the fitness value set *fv* and fitness difference set *fd*, as well as two features related to the improving neighbors set (*in*). Increasing the number of variable interactions increases the ruggedness of the landscape (i.e., flipping a few bits in the evaluated solution result in

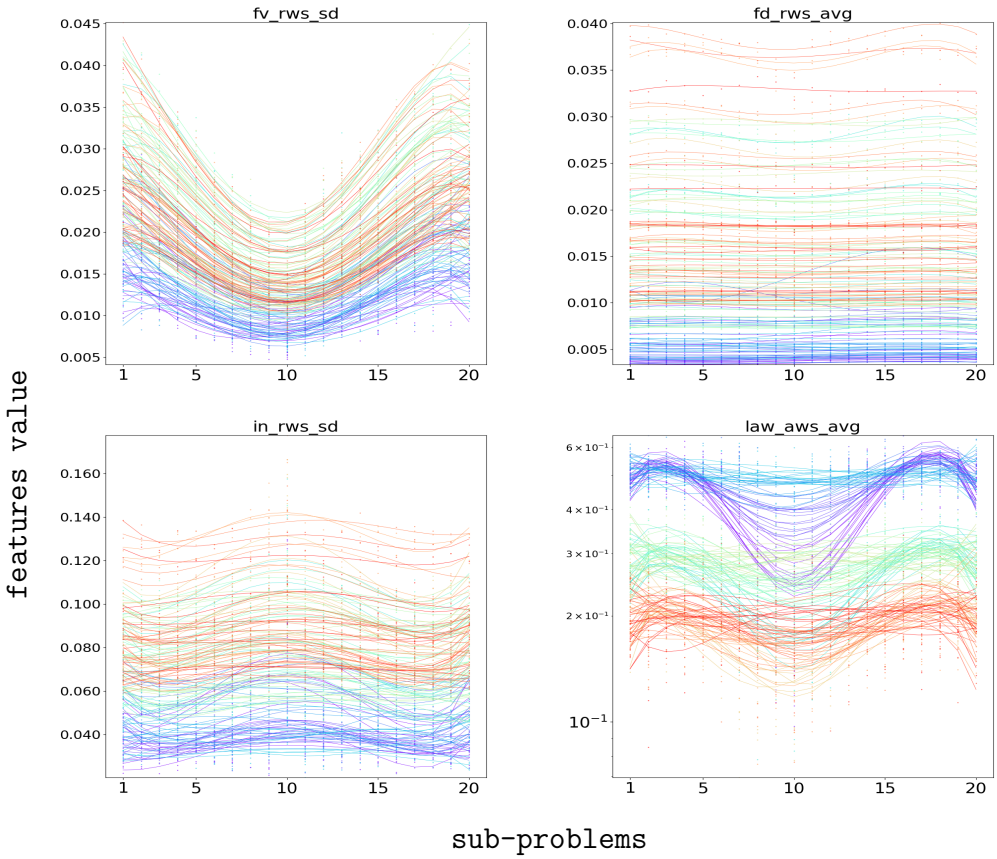


Figure 2.9: Feature values of $\rho$MNK-landscapes decomposed into 20 subproblems using a random walk or an adaptive walk for the length of the walk($n = 25$, each color correspond to a particular configuration of $\rho$ and $K$).

big change in the fitness of the new solution). Therefore, the fitness difference, which grasp the average difference of fitness between a solution and its neighbors, are related to $K$. The standard deviation of fitness or aggregating scalar features using standard deviation similarly characterize the epistasis $K$. The first coefficient of polynomial regression (*_c1_*) is particularly efficient to characterize the number of interactions between variables. Two features; computed with this function, (*fd_c1, in_c1*) are the most correlated with $K$. This statistic approximate the feature's growth on the adaptive walk. During the walk it becomes harder to find improving neighbors when $K$ is high as a small change in the actual solution may result of a large change in the fitness. Therefore, the mean and standard deviation of improving neighbors

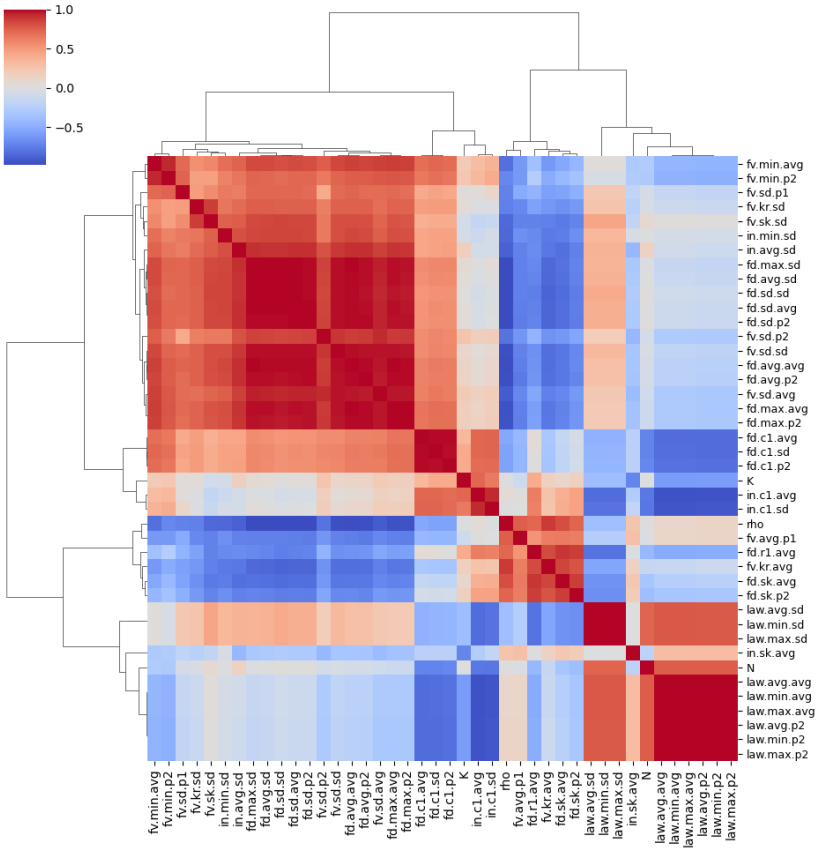


Figure 2.10: Correlation matrices between a subset of Decomposition-based features and parameter $\rho, K, n$ using independent samplings for the $\mu = 50$ weight vectors.

(*in_c1_avg, in_c1_sd*) both increase with $K$.

### Cluster associated with Objective Correlation

The second cluster of features (center of Figure 2.10) is strongly correlated to $\rho$. This cluster contains features from both fitness value *fv* and fitness difference *fd*. As the objective correlation impacts the shape of the Pareto front, polynomial coefficients of the mean fitness value, used to aggregate features, (*fv_avg_p1*) naturally belongs to this cluster. The most correlated features use statistics such as the kurtosis and the skewness (*kr, sk*). They grasp variation

of the shape of the Pareto front. Skewness capture the asymmetry of features distribution. A skewness of $sk = 0$ is equivalent to a (almost) symmetric distribution. When positively asymmetric $sk > 0$, the variation of values above the mean is larger than below the mean. Using an adaptive walk initialized with a random solution causes the positive asymmetry of the distribution, as only improving neighbors are selected. However, when objectives are correlated (i.e., positive $\rho$) solutions are concentrated and closer to the central weight $(0.5, 0.5)$. Thus, the difference in fitness and the asymmetry both increase. Similarly, the kurtosis shows the concentration of values around the mean. The higher the kurtosis value, the closer solutions are to the mean objective value. Given a negative $\rho$, the objective space is sparse between the objectives. Consequently, the range of objective values for a sub-problem is large. The parameter $\rho$ is also strongly correlated with the lengths of the adaptive walks. Intuitively speaking, anti-correlated problem instances have a larger front, so when using enough weights $\mu$ the variation of size between walks from the extremities is a meaningful feature to characterize $\rho$.

#### Cluster associated with Problem Dimension

The third cluster (bottom right of Figure 2.10) is correlated with the problem dimension $n$. It mostly contains features from the length of adaptive walk set ($law$). Although we consider the dimension as a parameter known by algorithms since the number of variables is necessary for their operation, we can see that the dimension of the problem is strongly correlated with the length of adaptive walks.

## 2.4 Conclusion

In this chapter, we defined the landscape of an optimization problem and why we need multi-objective features. We proposed a new methodology based on the concept of decomposition in order to define and to extract a number of new features. The approach is based on aggregating a set of single-objective feature computed with respect to a number of sub-problems obtained by decomposition in the objective space. We have shown that the so-designed features allows one to characterize the landscape intrinsic parameters, using $\rho$MNK-landscapes as a target benchmark.

At this stage of the presentation, there are however a number of open issues. Some parameters of the proposed approach can impact the feature values. Among these parameters, the weight vectors distribution, the sampling

method and the aggregation function are of specific interest. We will address these issues further in Chapters 3 and 4. In the following chapter, we use the designed decomposition-based features in order to address two high-level tasks. The first task consists in predicting the benchmark parameters, and the second task is a method to solve the algorithm selection problem. To better appreciate the accuracy of the decomposition-based multi-objective features, two others approaches for multi-objective landscape analysis are additionally considered.

# Chapter 3

# Landscape Analysis for High-Level Prediction Tasks

The contributions described in this chapter consist in using the decomposition-based features in order to solve two high-level tasks. The first task consists in predicting the benchmark intrinsic parameters. Specifically, in the case of $\rho$MNK-landscapes, we aim at predicting the two parameters $\rho$ and $K$ impacting the landscape. The second task consists in selecting the best algorithm from a portfolio for an unseen problem instance. In the adopted approach, we manage to first predict the performance of each algorithm, and we then select the best performing one. This task is known to apply for the Automated Algorithm Selection problem. Both of these tasks are termed as of a higher level, because they require the use of a machine learning models to make some predictions. In these models, features act as input to characterize the problem landscape. In this chapter, some results were published along with some results of Chapter 2 at the European Conference on Evolutionary Computation in Combinatorial Optimization (EvoCOP 2021): "Decomposition-based landscape analysis for multi-objective optimization".

## 3.1 Description of High-Level Tasks

### 3.1.1 Predicting Benchmark Parameters

In a black-box setting, the intrinsic structure of a problem is unknown. In the case of a problem generator such as $\rho$MNK-landscapes, the structure is tunable with two parameters: $\rho$ and $K$. The parameter $\rho$ controls the conflicting behavior between the objectives, and the parameter $K$ increases the ruggedness and multi-modality of the landscape. In Chapter 2, we showed that features are correlated with benchmark parameters. In the first high-level task we shall consider in this chapter, we push our analysis further by predicting the actual parameter values using the designed features.

The two landscape parameters we would like to predict are $\rho$ and $K$. The two other parameters $n$ and $M$ are assumed to be known. In fact, the dimension $n$ is necessary to construct the solutions, and the number of objectives $M$ can be easily known from the multi-objective nature of the problem. Thus, we consider only parameters directly affecting the landscape. The approach we adopt in the following is summarized in Figure 3.1. First, a set of problem instances are generated with different configurations of parameters $\rho$ and $K$. Secondly, landscape features are computed for each instance. Finally, two machine learning models are trained to predict $\rho$ and $K$, respectively.

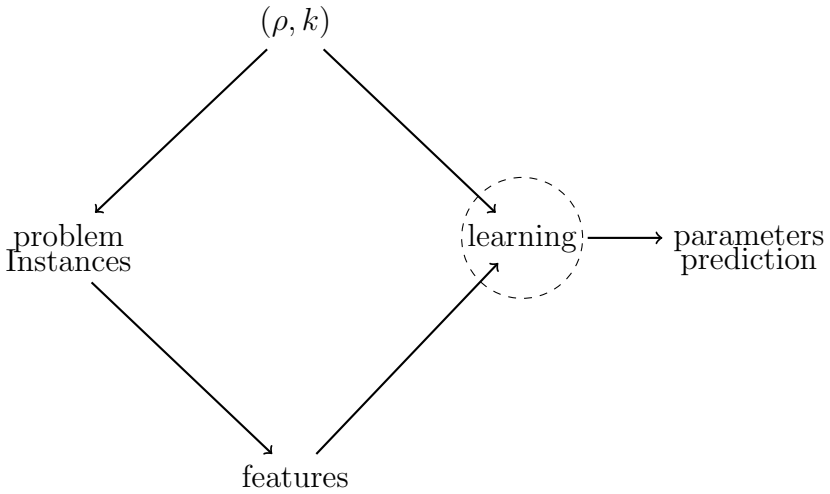


Figure 3.1: Schematic representation of the parameters' prediction task.

### 3.1.2 Automated Algorithm Selection

In the second task, we address the algorithm selection problem introduced by Rice in 1976 [77]. Let $\mathcal{P}$ be a set of optimization problem instances, and $\mathcal{A}$ a set of algorithms. Given an optimization problem instance $p \in \mathcal{P}$, the expected performance of the algorithm $a \in \mathcal{A}$ on $p$ is $\mathbb{E}(a, p)$. The algorithm selection problem is: find the algorithm $a_{best} \in \mathcal{A}$ such that

$$a_{best} = \underset{a \in \mathcal{A}}{\operatorname{argmax}}(\mathbb{E}(a, p)) \tag{3.1}$$

It is well known that no algorithm outperforms all others on all problems [99]. In the case of evolutionary algorithms, performance depends on chosen components and parameters, such as the crossover or the probability of mutation. In this work, we consider as performance indicator the average relative hypervolume deviation $\mathsf{rhv}(a, p)$ at a fixed budget. The hypervolume deviation is the normalized difference of hypervolume $\mathsf{hv}$ between an approximation of the Pareto front $S = a(p), a \in \mathcal{A}$ and the (best known) Pareto front. The exact Pareto front cannot be computed for large instances. Hence, we instead consider the approximation $\widetilde{PF}(p)$. Given a problem instance $p \in \mathcal{P}$, the Pareto front approximation is the set of all non-dominated solutions derived from the union of sets found by algorithms runs on the instance $p$.

$$\mathsf{rhv}(a, p) = \frac{\mathsf{hv}(\widetilde{PF}(p)) - \mathsf{hv}(a(p))}{\mathsf{hv}(\widetilde{PF}(p))} \tag{3.2}$$

In Figure 3.2, we summarize the main steps of the second task. The Ex-

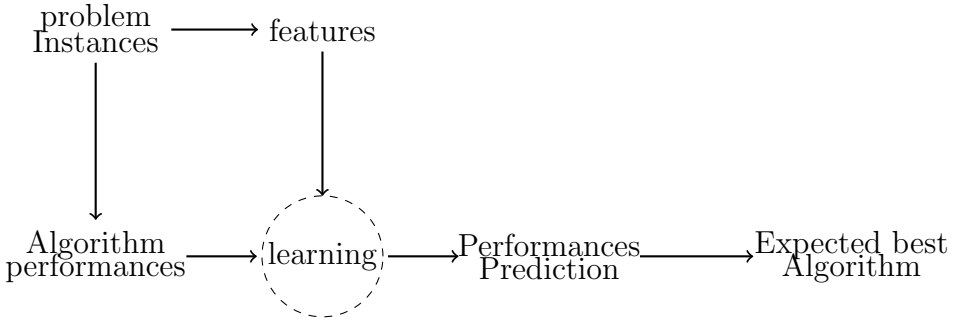


Figure 3.2: Schematic representation of the automated algorithm selection task.

ploratory Landscape Analysis approach, to solve the automated algorithm selection, has been considered by Mersmann [64] in the context of evolutionary algorithms and requires two sets. The first set, is a set of problem instances $\mathcal{P}$. For each instance $p \in \mathcal{P}$, landscape features are computed and are used as input in the learning model. The second set is a set of algorithms $\mathcal{A}$. For each algorithm $a \in \mathcal{A}$ and for each instance $p \in \mathcal{P}$, we compute the expected relative hypervolume deviation $\mathbb{E}(\mathsf{rhv}(a, p))$. Then, the model is trained to predict the expected relative hypervolume deviation of all algorithms given the features. The chosen algorithm is the one with the best predicted value.

The stopping condition of all algorithms is a fixed budget in terms of calls to the evaluation function. To predict the best performing algorithm, we could have considered other models. While we use the expected algorithm performance, there exists other approaches [11]. For example, the output can be the difference of performance between algorithms $\forall a_1, a_2 \in \mathcal{A}^2, a_1 \neq a_2, \mathbb{E}(a_1, p) - \mathbb{E}(a_2, p)$ or the output can be the rank of each algorithm on the problem $p$, etc.

## 3.2 Evaluating Prediction Models

### 3.2.1 Prediction Accuracy

In order to assess the prediction accuracy, we consider three measures. The first one is the coefficient of determination $R^2$. If the $R^2$ is low, i.e., $R^2$ closes to 0, this tells us that the input of the machine learning model does not explain well the output. On the contrary, when the $R^2$ is close to 1, the input data fits the prediction model. However, this measure has several weaknesses, in particular over-fitting could lead to a $R^2$ of 1. We also consider two alternative measures: the percentage of times the selected algorithm does not have the best hypervolume deviation in average, and the percentage of times the selected algorithm is statistically outperformed by at least one other algorithm. These two measures directly show the number of errors made by the prediction model. However, they are limited as the scale of error is not taken into account.

### 3.2.2 Accuracy of Automated Algorithm Selection

The merit indicator, which is a straightforward adaptation from [59], measures the gap between: (i) the performance of the single best solver (SBS) having the best performance on average over the training set (without model train-

ing), and (ii) the performance of the virtual best solver (VBS), obtained by a model that would make a perfect prediction. More precisely, let $I_{train}$ and $I_{test}$ be the set of training and testing instances, respectively, and let $\mathsf{rhv}(a, i)$ be the average relative hypervolume deviation of a given algorithm $a \in \mathcal{A}$ on instance $i \in I_{train} \cup I_{test}$. For every algorithm $a \in \mathcal{A}$ and instance subset $J$, let $\overline{\mathsf{rhv}}(a, J) = \frac{1}{|J|} \sum_{i \in J} \mathsf{rhv}(a, i)$. We define SBS as the algorithm having the best $\overline{\mathsf{rhv}}$ value on the training set $I_{train}$, i.e., $SBS = \text{argmin}_{a \in \mathcal{A}} \left\{ \overline{\mathsf{rhv}}(a, I_{train}) \right\}$. We define VBS as the *virtual algorithm* obtained by a perfect prediction model (an oracle); i.e., the algorithm with the best $\mathsf{rhv}(\cdot, i)$ value for each $i \in I_{test}$. Finally, let the recommended Solver (RS) be the algorithm predicted by the actual trained model. The *merit* indicator is computed as follows:

$$M = \frac{\overline{\mathsf{rhv}}(RS, I_{test}) - \overline{\mathsf{rhv}}(VBS, I_{test})}{\overline{\mathsf{rhv}}(SBS, I_{test}) - \overline{\mathsf{rhv}}(VBS, I_{test})} \tag{3.3}$$

It should be clear that: (i) a merit of 0 indicates that the model does not make any error, (ii) a merit in the range $[0, 1[$ indicates that the model is more efficient than the SBS but worse than the VBS, (iii) a merit greater than 1 indicates that the model is worse than the SBS. Achieving a merit value of 0 is clearly a very challenging task, and one seeks for a merit value below 1 (better than the SBS) and as close to 0 as possible (the VBS).

## 3.3 Experimental Setup

### 3.3.1 Sets of Features

In the previous chapter, we introduced a new set of decomposition-based features that we aim to analyze more deeply in this chapter. For completeness, we also consider analyzing our proposed features together with other existing features from the literature. Existing sets of features for landscape analysis fall into three categories, and these three categories are closely related to the paradigms used by multi-objective algorithms. The first set are features using the decomposition paradigm [104], introduced in the previous chapter. The second set are dominance-based features [55]. In this set, features computes statistics on the number of dominating, dominated and incomparable solutions locally in the neighborhood. Features from the third set are based on the use of multi-objective quality indicator [55]. Subsequently, these three different sets are respectively called decomposition-based, dominance-based and indicator-based feature sets.

**Decomposition-based features:** As introduced in Chapter 2, decomposition-based features are computed from a 3-step procedure. First, the multi-objective problem is split into $\mu$ single-objective sub-problems using the Chebyshev aggregation function. Then scalar features are computed for each of the $\mu$ sub-problems, and finally the features per sub-problems are aggregated. The previously introduced single-objective features [78] are: the **Fitness value (fv_*)**, the **Fitness difference (fd_*)**, **Improving neighbors (in_*)** and **Length of the adaptive walk (law_*)**. In addition to these scalar features, we propose an additional feature based on decomposition; but which is not inspired from the single-objective literature:

- **Maximal distance between improving sub-problems (spd_*)**. For every solution $x \in \mathcal{X}$, consider its neighbors $\mathcal{N}(x)$ and the ordered weight vectors $w_i, i \in \{1, \ldots, \mu\}$. We first compute for all neighbors in $\mathcal{N}(x)$ which sub-problems $f_{w_i}$ are improved, i.e., $g(x, w_i) < g_(x', w_i), x' \in \mathcal{N}(x)$. Then, the additional considered feature relates to the maximal distance between the improved sub-problems, normalized by the number of weight vectors $\mu$.

Notice, however, that this feature cannot be computed when using an independent sampling for each sub-problem (scalar walk) or when only one weight vector $\mu = 1$ is considered.

**Dominance-based features:** Dominance-based features use the dominance relation to compare solutions and their neighborhood [55]. We considered five subsets in the class of dominance-based features:

- The **Proportion of dominated neighbors (inf_*)** is the proportion of dominated neighbors for each solution in the sample.
- The **Proportion of dominating neighbors (sup_*)** is the proportion of dominating neighbors for each solution in the sample.
- The **Proportion of incomparable neighbors (inc_*)** is the proportion of incomparable neighbors for each solution in the sample.
- The **proportion of locally non-dominated neighbors (lnd_*)** is the proportion of solutions in the neighborhood of a given solution which are non-dominated.
- The **proportion of supported locally non-dominated neighbors (lsupp_*)** report the proportion of supported non-dominated neighbors.

A supported solution is a Pareto optimum whose objective vector belongs to the convex hull of the Pareto front [28]. When computed locally, given a solution $s$ and its neighbors $\mathcal{N}(s)$, a neighbor $s_i \in \mathcal{N}(s)$ is a supported solution if $s_i$ is not dominated and if $s_i$ is part of the convex hull computed on all non-dominated neighbors among $\mathcal{N}(s)$.

**Indicator-based features:** Indicator-based features use the hypervolume indicator to gather statistics on a given landscape [55].

- The **solution hypervolume (hv_*)** refers to the hypervolume between the fitness of a given solution and the reference point.
- The **neighborhood's hypervolume (nhv_*)** is the hypervolume between the set of neighbors $\mathcal{N}(s)$, for a given solution, $s$ and the reference point.
- The **hypervolume difference (hvd_*)** is the difference of hypervolume between a solution and each of its neighbors. For a given solution $s$, $hvd = \{hv(s) - hv(s_i) | \forall s_i \in \mathcal{N}(s)\}$.

For indicator-based features, we set the hypervolume reference point to the origin. The complete list of the dominance-based and indicator-based landscape features, with the detail of the used statistical tools, is available in Appendix 6.2.

### 3.3.2 Sampling Strategies for Landscape Analysis

Features computation require a sample of evaluated solutions. The scalar walk strategy, introduced in Chapter 2, computes independent sampling for each weight vectors, but it is not suitable for dominance-based and indicator-based features. These sets require a single sample for each problem instances. Thus, we considered a random walk and a multi-objective adaptive walk, as proposed initially [55] for dominance-based and indicator-based features. In the case of decomposition-based features, in addition to the scalar walk, we propose a method to compute features with a random and a multi-objective adaptive walk. Since random and adaptive walks are often used in the literature to compute features, the proposed method makes allows the use of the same sample to compute decomposition-based features with the two others sets.

In this method, the sample is computed independently of the decomposition process. For each solution of the sample, we compute the single-objective value

of all sub-problems $\mathtt{g}(\cdot, w^j)$. Then, for each sub-problem, we compute single-objective features. Finally, single-objective features of all sub-problems are aggregated, as described in Chapter 2. The major difference with the scalar walk method is that each solution is used in the features' computation of all sub-problems. All decomposition-based features are computed with the three methods: scalar walk, random walk and adaptive walk, but for a few exceptions. The first exception is the *length of adaptive walk* (law). It can only be computed with a multi-objective adaptive walk or a scalar walk with single-objective adaptive walks. It is also important to notice that the resulting feature is drastically impacted by the sampling method. The second exception is the *maximal distance between improving sub-problems* (spd), which can only be computed with a random or an adaptive walk.

### 3.3.3 Machine Learning Models

Several techniques and machine learning methods for the task of selecting algorithms have been applied in the literature [11]. Some models propose to predict the ranking of algorithms according to their respective performance, others models predict the difference in performance between algorithms or directly predict the performance value of each algorithm. Learning methods include regression trees, or random forests. As mentioned earlier, there exists other approaches which predict directly the best algorithm, hence classification trees can also be used. It was shown at the *Open Algorithm Selection Competition* [59] that a machine learning model's accuracy is related to the prediction scenario, i.e., the input and the output data. In the following, we chose to use random forest regression [80] to predict algorithm performance because it was shown to work relatively well in previous studies [95, 79, 20, 11]. Additionally, it has the advantage of easily returning importance measures on the features used in input, providing insights of the relation between features and algorithm performance.

### 3.3.4 Algorithm Portfolio

As mentioned in Chapter 1, the MOEA/D algorithm is based on a flexible decomposition-based framework that can be configured in different manners. In its baseline variant [104], MOEA/D first decomposes the problem into a number of scalarized sub-problems, as discussed previously. Then, a solution is evolved for each sub-problem in a cooperative way. The algorithm iterates over the sub-problems and, at each iteration, an offspring is generated by means of crossover and mutation on the basis of parent solutions selected from the so-

called $T$-neighborhood; i.e., the sub-problems corresponding to the $T$ closest weights in the objective space. The new offspring can then replace any of the sub-problem solutions in the $T$-neighborhood of the current sub-problem. This corresponds to a standard evolutionary process, where selection and replacement are performed iteratively over sub-problems. In [62], it is shown that the selection and replacement underlying the standard MOEA/D framework are key algorithm components that highly impact its performance. Several generational variants are proposed therein, allowing to tune the selection and replacement underlying the MOEA/D framework from fully cooperative (i.e., among *all* sub-problems) to fully selfish (i.e., independently of any other sub-problem).

Interestingly, it was found that no variant outperforms the other independently of the benchmark parameter values $\rho$ and $K$ for the considered $\rho$MNK-landscapes. Since such parameters are unknown in a black-box optimization scenario, the study presented in [62] leaves open the challenging question of which variant to choose in an automated manner. In addition, this constitutes a perfect and typical setting for the main automated algorithm selection task addressed in this chapter. In the following, We consider three representative variants of the MOEA/D framework, exposing different degrees of cooperation among sub-problem solving. These variants are denoted as follows: (i) MOEA/D referring to the standard variant [104], (ii) MOEA/D-SC, a generational variant where selection is performed selfishly for every sub-problem whereas replacement is performed cooperatively, and (iii) MOEA/D-SS, a (selfish) generational variant exposing the lower degree of cooperation among sub-problems. Besides population size, the three variants have the same set of parameters. The parameter $\delta = 1$ asserts that parents solutions belong to the $T_i$ closest sub-problems. Parameter $nr = 2$ indicates the maximal number of replacement for each offspring. The mutation probability is set to $p_{mut} = \frac{1}{n}$, and we used a single-point crossover $p_{cr} = 1$.

In order to highlight the relevance of this portfolio in studying the accuracy of our features when integrated into a landscape-aware algorithm selection approach, we briefly report their relative performance using exactly the same set of $\rho$MNK-landscapes as in the previous chapter. Every algorithm is executed 20 times on each instance, using a population size equals to $n$, and a budget of $10^6$ evaluations. The performance of an algorithm is computed as its hypervolume relative deviation w.r.t. the best-found approximation set for each instance. For a given instance, the hypervolume reference point is set to the worst value seen across all runs for each objective. We then count the number of times an algorithm is statistically outperformed using a two-sided Mann-Whitney test with a p-value of 0.05 with Bonferroni correction. Results

Table 3.1: Performance matrix of the three MOEA/D variants. The diagonal in gray reports the number of times the corresponding algorithm is statistically outperformed by another one (the lower, the better). The other cells report how many instances (out of 1000) the algorithm in the corresponding line is statistically better than the algorithm in the corresponding column (the higher, the better).

| | MOEA/D-SC | MOEA/D | MOEA/D-SS |
|---|---|---|---|
| MOEA/D-SC | 205 | 18 | 310 |
| MOEA/D | 85 | 137 | 312 |
| MOEA/D-SS | 120 | 119 | 622 |

are reported in Table 3.1.

We clearly see that each algorithm is outperformed by another one on a subset of instances. The basic MOEA/D variant seems to have a reasonably good behavior, since it is less often outperformed than the two other variants overall (see diagonal). We also observe that there is a complex interaction between algorithm performance and the benchmark parameters $\rho$ and $K$: (i) the smaller $K$ and $\rho$, the better MOEA/D and MOEA/D-SC against MOEA/D-SS, (ii) the larger $\rho$ (highly correlated), the better MOEA/D-SS, and (iii) the larger $K$ and the smaller $\rho$, the better MOEA/D-SC. Of course, this general trend has some exceptions, but it shows the impact of the unknown benchmark parameters on the relative performance of algorithms.

## 3.4 Results and Discussions

### 3.4.1 Predicting Benchmark Parameters

To investigate the accuracy of decomposition-based features, we consider a typical machine learning task consisting in predicting the value of the (unknown) global benchmark parameters $K$ and $\rho$. We respectively construct a random forest regression model [15] for $K$ and $\rho$, using the whole set of multi-objective features computed over all considered $\rho$MNK-landscapes. Random forest has the nice property of providing a measure of feature importance for model fitting. In Table 3.2, we report the average $R^2$ obtained by models trained with features. Features have been computed a scalar walk, a random walk and a multi-objective adaptive walk (dominance walk), respectively. The set of features computed with a scalar walk consist of the union of the three subset of features fitness difference, fitness value and improving neighbors com-

Table 3.2: Average $R^2$ obtained by random forest regression models trained to predict benchmark parameters using decomposition-based features as input. Features are computed respectively from a scalar walk, a random walk and a dominance walk.

| | Scalar Walk | | | Random Walk | Dominance Walk |
|---|---|---|---|---|---|
| | $\mu = 3$ | $\mu = 20$ | $\mu = 100$ | | |
| Parameter $\rho$ | 0.95 | 0.97 | 0.93 | 0.96 | 0.95 |
| Parameter $K$ | 0.91 | 0.96 | 0.91 | 0.97 | 0.85 |

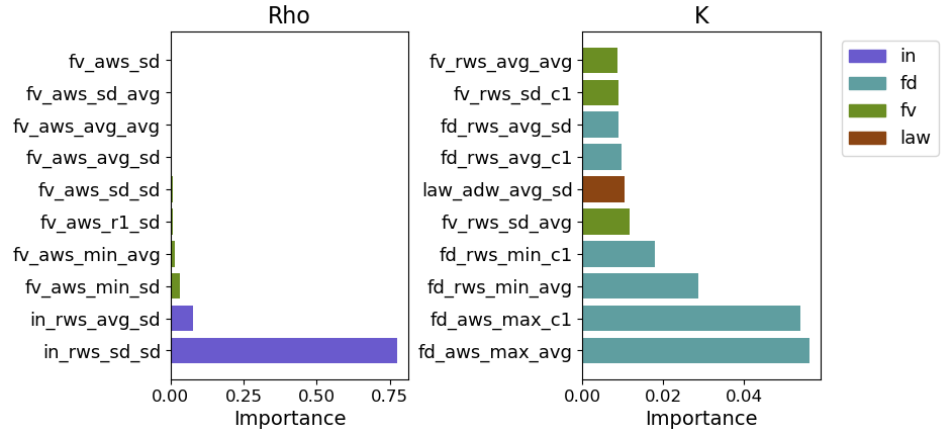


Figure 3.3: Relative importance of decomposition-based features to predict $\rho$ (left) and $K$ (right) for features computed with a random walk (*_rws_*) and with an adaptive walk(*_aws_*). Only the ten most important features are reported.

puted with a random scalar walk and the length of the walk computed with an adaptive scalar walk. In Figure 3.3, we report the relative importance of each feature extracted from the random forest models, using the Gini impurity as a measure of quality. Values are averaged over 10 independent repetitions of model fitting. Unlike Table 3.2, in Figure 3.3 models are trained with the union of features computed from a random walk and features computed from an adaptive walk.

Every sampling strategy reaches a relatively high $R^2$ score. However, the dominance walk and the scalar walk with a low value of $\mu = 3$ or a high value of $\mu = 100$ have a lower $R^2$ score than the random walk or a scalar walk with $\mu = 20$. The first notable observation is that feature importance, depicted in Figure 3.3, is different depending on whether we aim at predicting the

benchmark's parameter $\rho$ or $K$. The objective correlation $\rho$ appears mostly related to a single feature: the standard deviation of the number of improving neighbors (in_rws_sd). By contrast, the parameter $K$ is related to multiple different features, that mostly correspond to the fitness difference computed from dominance walk, in particular the mean fitness difference (fd_aws_max_avg).

### 3.4.2 Landscape-aware Algorithm Selection

In this section, we conduct a second set of experiments in order to study the accuracy of the designed features when integrated into an automated algorithm selection approach. We consider the more sophisticated task of selecting the best performing algorithm among an algorithm portfolio (second task).

**Experimental setup:** We study the accuracy of the decomposition-based features by investigating the selection of the best performing MOEA/D variant. For this purpose, we adopt the following standard supervised-learning approach. We first train three models in order to predict the performance of every considered MOEA/D variant. We use the average hypervolume deviation as defined in the previous section as a measure of algorithm performance, which then corresponds to our output prediction variable. Considering an unseen test instance, the landscape features are first computed, the performance of each algorithm is then predicted on the basis of the trained models, and the algorithm having the best prediction is selected as the recommended one. We consider the same set of $\rho$MNK-landscapes described in the previous section. We adopt a standard validation methodology where an instance is selected for training with probability 0.9 and for testing with probability 0.1. We use a set of 100 random regression trees to learn and predict the expected relative hypervolume deviation. Reported values are computed over 50 independent folds.

We recall that the proposed multi-objective features rely on some weight vectors $\mu$. We consider a variable number of weight vectors in the range $\mu \in \{1, 2, 3, 4, 5, 6, 10, 20\}$. Additionally, we consider two alternatives for generating weight vectors, namely uniform or random. In a random setting, a weight vector is generated uniformly at random. In a uniform setting, the weight vectors are evenly distributed in the objective space. In particular, for $\mu = 1$, the weight vector selected in the uniform setting is $(0.5, 0.5)$; i.e., the "middle" of the objective space. In this case, the multi-objective features are simply the same as the corresponding scalar features from the single scalarized sub-problem. For $\mu = 2$, our uniform setting corresponds to weight vector $(1, 0)$ and $(0, 1)$. This means that our features are obtained by aggregating

the scalar features computed independently for each objective of the original multi-objective problem, similar to others study from the literature [50]. The impact of this setting is carefully analyzed in our experiments.

**Impact of the weight vector distribution:** Our main results are summarized in Figure 3.4, showing the prediction accuracy as a function of the number of weight vectors $\mu$ and their type (random or uniform). For completeness, we also show the $R^2$ coefficient obtained by the training models. Different observations can be extracted from Figure 3.4.

First, we clearly see that the choice of the weight vector distribution is of critical importance. In fact, a random choice does not obtain a good accuracy, except when the number of weights $\mu$ is substantially large. By contrast, a uniform strategy appears to perform reasonably well, even when the number of weights is low. Interestingly, for uniform weights, the worst accuracy is obtained with $\mu = 2$. Notice that such a setting is even substantially outperformed by a random choice of weight vectors. This indicates that computing scalar features independently for each objective is not a recommended strategy. By contrast, computing features for decomposed sub-problems is effective even when using a very low number of weights. This indicates that a decomposition-based approach for multi-objective landscape analysis contains valuable information about algorithm performance. Surprisingly, we found that a uniform choice of few weight vectors with $\mu \in \{1, 3\}$ performs reasonably well, although increasing $\mu > 3$ allows to obtain a better accuracy. The relatively good results achieved with $\mu = 1$ are however to be interpreted very carefully, taking into account that the shape of the Pareto front for $\rho$MNK-landscapes, although having different magnitude in the objective space, is roughly convex, symmetric, and centered in the middle of the objective space, regardless of the values of $\rho$ and $K$ [95]. Although this is a recurrent observation for many multi-objective combinatorial optimization problems, one might need to carefully choose $\mu$ when tackling problems with different Pareto front shapes. Such considerations are left for future investigations.

At last, in order to further show the accuracy of the proposed multi-objective landscape features, we experiment a baseline random forest model using the (unknown) global benchmark parameters $\rho$ and $K$ as input variables to predict algorithm performance. By contrast with a black-box scenario where the knowledge about $\rho$ and $K$ is not available, the accuracy of such a 'white-box' model should highlight the relevance and reliability of the proposed black-box features. We found that such a model trained with $\rho$ and $K$ obtains an average merit of 0.41. Comparatively, black-box features obtain

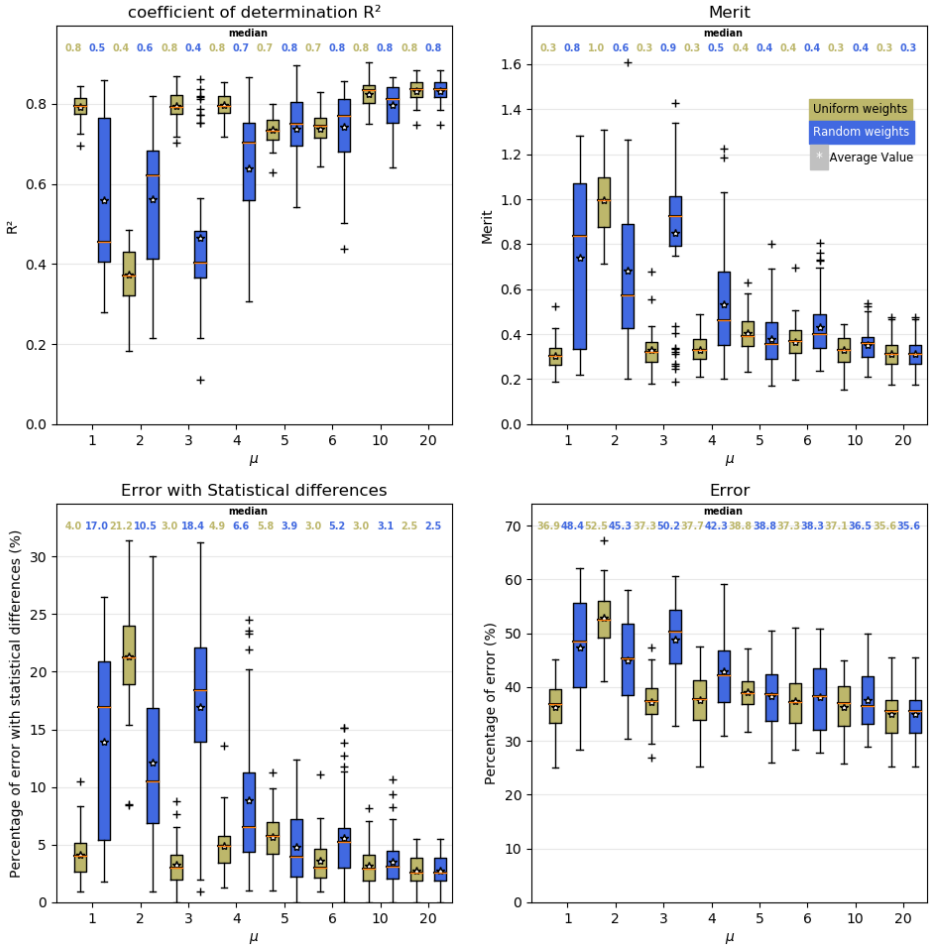


Figure 3.4: Merit (top right), $R^2$ (top left), Error rate of selecting a non-statistically better algorithm (bottom left) and Error rate of not selecting the best algorithm (bottom right) according to the number of weight $\mu$ and their distribution.

an average merit of 0.31, 0.37 and 0.29 respectively, for $\mu \in \{1, 3, 20\}$ uniform weight vectors. This indicates that the proposed approach is very effective, and that the designed high-level black-box features seem to provide more accurate prediction models than the global benchmark parameters, hence allowing to substantially improve over the single best solver, and also to get closer to the ideal virtual best solver.

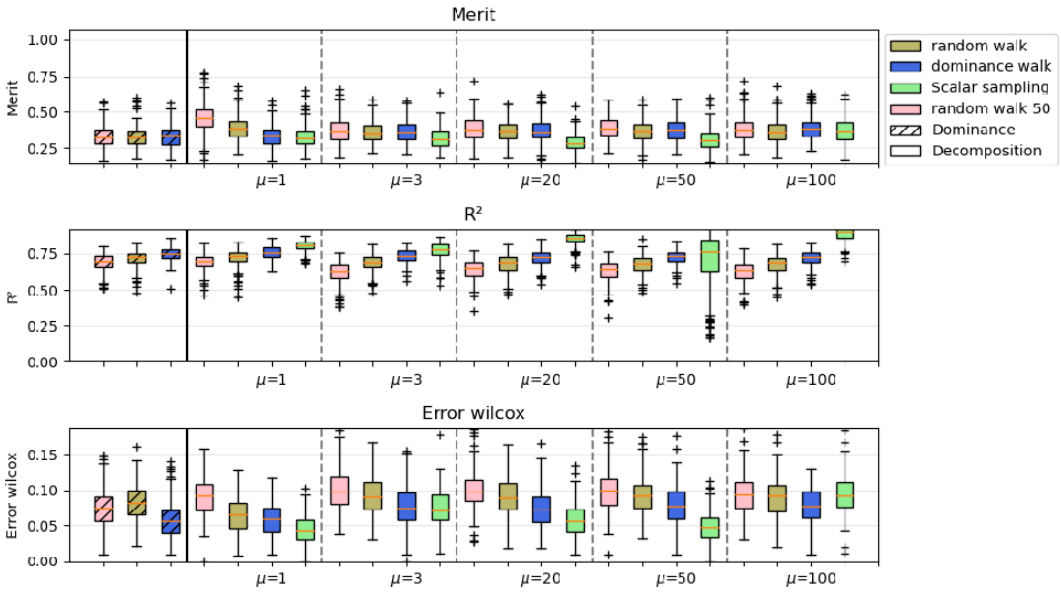


Figure 3.5: Merit (top), $R^2$ (center), Error rate of selecting a non-statistically better algorithm (bottom) according to the number of weights $\mu$ for decomposition-based features with all sampling methods and dominance features (hashed boxes).

**Impact of the number of weight vectors:** Reaching an $R^2$ score greater than 0.75, decomposition-based features computed from a scalar walk (*g*reen box in Figure 3.5) are suitable to predict algorithm performance. As $\mu$ increases, the score increases, reaching an $R^2$ of 0.90 for $\mu = 100$. Surprisingly, when $\mu = 50$ the distribution of the values of $R^2$ is large, with some cases below 0.25. The regression score shows that most of the variation between problem instances are covered by landscape features. However, it does not correlate with other accuracy measures. In particular, when $\mu = 100$, we observe the highest $R^2$ score but according to other measures, features are less accurate.

The accuracy of the machine learning model reach an error rate with statistical difference around 5% for $\mu \in \{1, 3, 20, 50\}$. Meaning that only 5% of the prediction fail to predict a statistically better algorithm. As a comparison, if the learning model always selects the best algorithm in average (MOEA/D), the statistical error rate would be 13.7% and the percentage of error (predicting exactly the best algorithm without statistical equivalence) would be 64%. It implies that using the selected algorithm is often a better choice than using

Table 3.3: Average decomposition-based features subset importance using the Gini impurity as a measure of quality computed from a scalar sampling. The gray row achieved the best result when comparing the average Merit.

| $\mu$ | law | fv | fd | in |
|---|---|---|---|---|
| $\mu = 1$ | 0.0994 | 0.1923 | 0.2747 | 0.4335 |
| $\mu = 3$ | 0.1059 | 0.3119 | 0.1709 | 0.4112 |
| $\mu = 20$ | 0.0963 | 0.2175 | 0.2362 | 0.4499 |
| $\mu = 50$ | 0.5130 | 0.1378 | 0.1332 | 0.2153 |
| $\mu = 100$ | 0.6721 | 0.0793 | 0.0733 | 0.1752 |

the best algorithm on average. However, a possible source of error for the machine learning model is that both MOEA/D and MOEA/D-SC stand out when $\rho < 0.25$ and no statistical difference was made in this configuration.

Using decomposition-based features with a scalar sampling is efficient, reaching a merit of 0.289 for $\mu = 20$ meaning that using the selected algorithm instead of the single best algorithms (SBS) will reach a difference of hypervolume three time smaller than the SBS. In the same way as for the other tools to evaluate the model ($R^2$, error rate), the merit varies according to $\mu$, improving gradually when $\mu$ goes from 1 to 20 except for $\mu = 100$ where it is only at 0.37. This indicates that the decomposition-based approach with a *scalar walk* is effective. The selected algorithm allows to substantially improve over the single best solver and to get closer to the ideal virtual best solver.

**Importance of Decomposition-based Features:** Table 3.3 shows the evolution according to $\mu$ of the average importance of each decomposition-based feature's subset, using the Gini impurity as a measure of quality. All subsets are useful according to $\mu$. As expected in the previous correlation analysis, the length of adaptive walk *law* provides very little information when $\mu$ is too small ($\mu \leq 3$). On the other hand, when $\mu$ grows, its importance in the prediction is greater, up to 67% for $\mu = 100$. However, in this case, the merit obtained is worse, and therefore the prediction is of poorer quality. The number of improving neighbors *(in)* is the most stable subset. It is useful for any value of $\mu$ varying from 17% to 45%. The last two subsets, fitness value *fv* and fitness difference *fd* have similar roles in regressions, they have a high importance between 17% and 31% when $\mu$ is low. However, when $\mu \in \{20, 50, 100\}$, the length of the adaptive steps increases in importance, and the combined importance of *fv* and *fd* fall below 15% when $\mu = 100$.

Table 3.4: Mean Merit for Dominance, indicator and decomposition-based landscape analysis according to the number of weight $\mu$ and the sampling strategy.

| | | Merit | | | |
|---|---|---|---|---|---|
| $\mu$ | Set | rnd $\ell = 50$ | rnd $\ell = 100$ | dom | scalar |
| *None* | Dominance ∪ Indicator | 0.327 | 0.331 | 0.334 | |
| $\mu = 1$ | Decomposition | 0.459 | 0.390 | 0.338 | 0.332 |
| $\mu = 3$ | Decomposition | 0.373 | 0.362 | 0.366 | 0.332 |
| $\mu = 20$ | Decomposition | 0.388 | 0.367 | 0.372 | **0.289** |
| $\mu = 50$ | Decomposition | 0.391 | 0.371 | 0.380 | 0.315 |
| $\mu = 100$ | Decomposition | 0.386 | 0.370 | 0.386 | 0.370 |

**Impact of the sampling method for decomposition-based features:** We computed decomposition-based features from a dominance walk and random walk (of size 50 and 100). All merit values computed on empirical performance models with a k-fold cross validation are presented in Table 3.4 as well as their relative rank in Table 3.5. Additionally, Figure 3.5 highlights the difference in accuracy for each sampling method.

Generally speaking, decomposition-based features with other sampling methods than scalar walk are of lower quality, reaching in the best case a merit of 0.327. This difference in performance is at first observed via the $R^2$ score, which is constantly dominated by features computed with a scalar walk. When a dominance walk is used, the merit decreases with $\mu$. When $\mu = 1$ the error rate and the merit values are comparable to a scalar walk with $\mu = 1$. This result can be explained by the similarity between a dominance walk and a single-objective adaptive walk on the weight vector $(0.5, 0.5)$ equidistant from the two objectives. Random walks are less efficient. Intuitively, the size of the random walk is a key factor in feature precision. This is confirmed in our results, since $\ell = 50$ performs worse than $\ell = 100$. Although the comparison between scalar walk, dominance walk and random walk shows that decomposition-based features are more efficient when computed from a scalar walk, all methods manage to predict efficiently the best algorithm. Achieving a merit of 0.338 for $\mu = 1$ with adaptive walk, and on average a merit $M < 0.4$, using the machine learning model to select the most suited algorithm outperform the best algorithm in average.

Table 3.5: Features rank obtained from the Merit distribution for all set of features according to $\mu$ and the sampling strategy.

| | | Rank method | | | |
|---|---|---|---|---|---|
| $\mu$ | Set | rnd $\ell = 50$ | rnd $\ell = 100$ | dom | scalar |
| *None* | Dominance ∪ Indicator | 2 | 2 | 3 | |
| $\mu = 1$ | Decomposition | 22 | 16 | 3 | 2 |
| $\mu = 3$ | Decomposition | 8 | 8 | 8 | 1 |
| $\mu = 20$ | Decomposition | 16 | 8 | 8 | 0 |
| $\mu = 50$ | Decomposition | 16 | 8 | 9 | 1 |
| $\mu = 100$ | Decomposition | 15 | 8 | 14 | 8 |

### 3.4.3 Comparison with Existing Multi-objective Features

Dominance and indicator-based features are extremely competitive with the decomposition-based features computed with a scalar walk (Tables 3.4 and 3.5). We consider training machine learning models with the union of dominance and indicators features. In Figure 3.5, dominance and indicator-based features are the three hashed boxes on the left computed with a random walk of size 50, a random walk of size 100 and a dominance walk, respectively in color pink, brown, and blue. Results are similar, with decomposition-based features reaching an average merit between 0.327 and 0.334. Dominance and indicator-based features are also less restrictive in terms of the choice of sampling method. Whether it is a random walk or a dominance walk, the results are stable and of the same order of magnitude. Although the gap in accuracy is small, our results show that using decomposition-based features with a scalar sampling and a well configured $\mu$ can outperform dominance and indicator features. This behavior is particularly pronounced in the ranks in Table 3.5. Decomposition-based features for $\mu \in \{3, 20, 50\}$ are ranked 0 and 1, meaning that it statistically outperforms dominance-based and indicator-based features, which rank 2 at best.

## 3.5 Conclusion

In this chapter, we have investigated in more depth the accuracy of the proposed set of decomposition-based features. Additionally, we proposed an alternative method to extract decomposition-based features with the most common sampling strategies. To improve our understanding of decomposition-based features, we performed two experiments: (i) predicting benchmark parame-

ters and (ii) predicting the best algorithm for a given problem instance. In these experiments, we studied how feature accuracy varies according to weight vectors and to the sampling method. We showed that decomposition-based features can efficiently predict benchmark parameters and predict the best performing algorithm. Nevertheless, weight vectors used in the feature extraction is a critical component. Firstly, if the distribution of weight vectors is not suitable, the accuracy decreases dramatically. The second critical component in decomposition-based features is the choice of the number of weight vectors. While features perform properly in average, this is only when the number of weight vectors is correctly configured that decomposition-based features showed to be particularly accurate. In addition, among the different decomposition-based features, all features proved to be useful in the machine learning model.

To push our analysis further, we also considered two other sets of features: dominance-based features and indicator-based features. We showed that the difference in accuracy between feature sets is relatively small. If a random or a dominance walk is used in the computation, models trained with decomposition-based features are comparable to dominance-based or indicator-based features. When using a well configured scalar walk, decomposition-based features allowed machine learning models to be slightly more accurate.

Until now, we considered conducting the landscape analysis to be cost-free. It is however not the case in practice, since the computation of features requires a sample of solutions, and this sample requires additional function evaluations. Accordingly, in the next chapter, we will address the cost of sampling on features accuracy in higher-level tasks. This will include methods to reduce the sampling size and how features are impacted by such reductions. Finally, an extension to the merit measure defined in this chapter is presented in order to accurately measure the gain when predicting and selecting the best performing algorithm while taking the cost of feature computation into account.

# Chapter 4

# Feature Cost Analysis

The design of effective features enabling the development of automated landscape-aware techniques requires addressing a number of inter-dependent issues. In this chapter, we are interested in contrasting the amount of budget devoted to the computation of features with respect to: (i) the effectiveness of the features in grasping the characteristics of the landscape, and (ii) the gain in accuracy when solving an unknown problem instance by means of a feature-informed automated algorithm selection approach. We study simple cost-adjustable sampling strategies for extracting different state-of-the-art features. Based on extensive experiments, we report a comprehensive analysis on the impact of sampling on landscape feature values, and the subsequent automated algorithm selection task. In particular, we identify different global trends of feature values leading to non-trivial cost-vs-accuracy trade-off(s). Besides, we provide evidence that the sampling strategy can improve the prediction accuracy of automated algorithm selection. Importantly, this holds independently of whether the sampling cost is taken into account or not in the overall solving budget.

The contributions presented in this chapter are part of the paper "Cost-vs-Accuracy of Sampling in Multi-objective Combinatorial Exploratory Landscape Analysis" published and presented at the Genetic and Evolutionary Computation Conference (GECCO 2022) [22], with minor extensions.

## 4.1 Motivations and Methodology

The combination of landscape features extraction with the development of automated recommendation systems proved to be of importance for state-of-the-art black-box optimizers. As observed in Chapter 3, algorithms expose different performances as a function of the problem instance being solved. Multi-objective features provide meaningful information of the (black-box) landscape underlying a given instance. Following a standard supervised machine learning methodology, a model can then be trained on the basis of extensive experiments on a set of known instances. The training phase leads to a mapping of (pre-computed) landscape feature values to the performance of (pre-executed) algorithms for which the landscape features are pre-computed and the available algorithms are executed, in order to elicit their performances. Given a new unseen instance, features can thus be computed and provided as input to the trained model in order to obtain a prediction about the most suitable algorithm for solving the instance. A key ingredient for the success of such approach relies on the ability to design cheap and meaningful features. In this chapter, we are interested in studying the impact of feature extraction for multi-objective combinatorial optimization and when applying in higher-level tasks.

Extracting black-box features requires a particular sampling of solutions from the search space. This consists in evaluating the objective values of a carefully-chosen set of solutions, on the basis of which a numerical statistic, constituting the feature value, is computed. The sampling has two critical implications. First, the feature value can be different depending on the sampling method [76] and the *size* of the sampling. This has a direct consequence on the quality of the machine learning model, and hence on the accuracy of an automated algorithm selection approach. Second, to tackle the algorithm selection problem with machine learning, one has to accommodate the cost of sampling, since the sample size impacts the overall budget, in terms of the number of evaluations affordable to solve a new unseen instance. Consequently, computing features that are both as informative as possible, and as cheap as possible, is a critically important issue. This chapter aims at pushing a step towards a better understanding of the impact of the sampling type and cost on the design of a successful automated algorithm selection approach. In particular, this questions about the impact of the sampling cost on feature values, and rises a number of questions concerning the accurate choice of the type of sampling method and the sample size.

### 4.1.1 Background

In single-objective continuous optimization, eliciting the impact of sampling is relatively well studied. On the one hand, different sampling techniques such as uniform sampling, Latin hypercube, or Sobol sequence were studied in the past [51, 65, 61]. Very recently, it was shown that, quoting the authors in [75, 76], "feature values are not absolute ... [and] cannot be interpreted as stand-alone measures" independently of the method being used for sampling. Such a statement was supported by an analysis of the high-level properties of features, such as their expressiveness and their robustness, with respect to sampling. On the other hand, the computational effort needed for feature computation was studied in the past [81, 49, 75, 76, 58]. For instance, recommendations for computing cheap features are given in [49]. Reviewing all the literature is out of the scope of this chapter, since our focus is on multi-objective combinatorial optimization, but let us however remark that very few features can be interchangeably considered for continuous and discrete domains. In fact, sampling in discrete domains is fundamentally different from sampling in continuous domains, and so are the existing features.

Relying on a sampling method is mandatory to explore the landscape. As commented previously in different part of this thesis, the most common sampling technique is to perform some particular *walk* over the landscape [96, 46, 78]. Roughly speaking, this is intended to inform about the challenges that search algorithms have to face when exploring the landscape underlying a black-box optimization problems, such as the landscape ruggedness or the distribution of local optima. Although we can find a number of analysis [90, 89, 93] with respect to feature sampling in *single-objective* combinatorial optimization, very few lessons can be learned when it comes to integrate those features in an effective automated algorithm selection approach. In *multi-objective* combinatorial optimization, which is our main focus, few studies can be reported. In a state-of-the-art feature-based approach [55], the cost of sampling is shown to represent a relatively low proportion of the overall budget dedicated to running the automatically-selected algorithm. In [57], a similar observation is also reported. Despite such observations, the question of how the sampling influences the accuracy of a feature-based automated algorithm selection remains open.

### 4.1.2 Cost of Features

In Chapter 3, we have actually ignored the cost of feature computation. This cost is mostly dependent on the considered sampling method used to extract

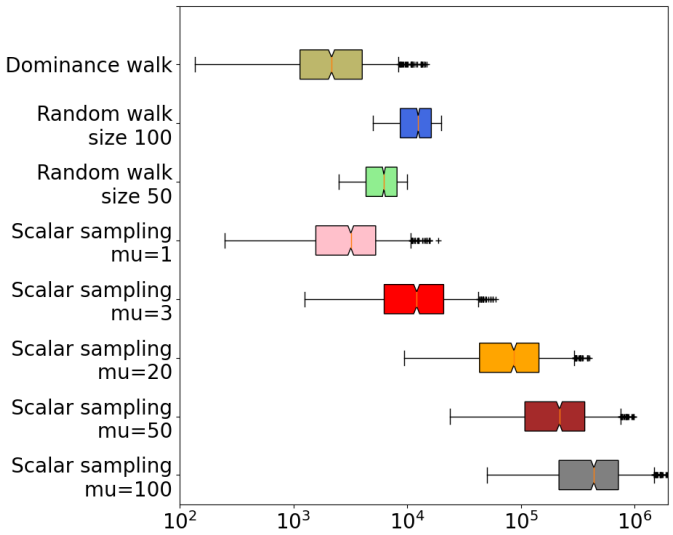


Figure 4.1: Number of calls to the evaluation function required for each sampling methods on $\rho$MNK-landscapes with $n \in \{50, \ldots, 200\}$.

the feature value. To illustrate, in Figure 4.1, we show the number of function evaluations needed for the feature computation with respect to all sampling methods considered in Chapter 3 (including all values of $\mu$). As a reminder, in all walks, when a solution is visited, all of its neighbors are evaluated. Let us remind that algorithm performance was computed with a fixed budget of $10^6$ function evaluations. In the automated algorithm selection, the predicted performance is therefore fixed for the same budget. Decomposition-based features computed with an adaptive scalar walks have a cost related to the number of weights vectors $\mu$. Intuitively, an adaptive walk is computed for each weight independently, this lead to a drastic increase in cost as $\mu$ grows. When $\mu = 1$, the average cost per problem is above $10^3$, equivalent to 0.1% of the algorithm budget. When $\mu = 3$ the average cost increases by a factor of 3, reaching 1% of the budget and for $\mu = 20$, the cost reaches $10^5$. The remaining two values, $\mu = 50$ and $\mu = 100$, are so expensive that the sampling consumes almost all the budget or even exceeds the budget with an average size of 35% and 70% of the number of function evaluations allowed to be computed. In comparison, common sampling methods such as random walk and dominance walk require less function evaluations. Random walks with a tunable budget (i.e., of size 50 and 100 in previous experiments) used roughly $10^4$ evaluations, equivalently

to 1% of the algorithm budget. It also has the advantage of having a cost only related to the problem instance dimension $n$. Finally, the dominance walk has a similar cost than a scalar adaptive walk with $\mu = 1$. It varies between $10^3$ and $10^4$. Among the previously used sampling strategies, the most expensive was the scalar walk when $\mu$ is large, reaching a cost between 10% and 100% of the budget for algorithms runs. Thus, it is clear that when incorporating the features cost into an automated algorithm selection approach, we should carefully take into account the sampling cost.

### 4.1.3 Methodology

In the following, we provide the first in-depth investigations on the impact of sampling in multi-objective combinatorial landscape analysis. We rely on the fact that the computation of multi-objective landscape features are mainly influenced by: the type of the walk, the length of the walk, and the explored portion of the neighborhood. We hence propose to conduct a systematic empirical analysis of the impact of each factor, with a particular focus on the sampling cost issues. More precisely, our contribution can be summarized as follows:

- Considering two conventional sampling methods from the literature, we adjust the amount of budget devoted to the feature computation by: (i) constraining the number of neighbors to be evaluated along the walk, and (ii) controlling the length of the walk when the sampling method allows us to do so. This departs from the usual way of computing landscape features, where it is commonly believed that solely the overall amount of budget can have an impact on feature values. Such settings are then analyzed using an extensive number of features taken from the two state-of-the-art sets: dominance- and indicator-based features [55], and the decomposition-based features set [21] proposed in Chapter 2.

- These cost-adjustable features are analyzed in a fine-grained manner by eliciting their correlation with the global problem characteristics, and in a coarse-grained manner when integrated within an automated algorithm selection approach. This allows us to contrast the amount of budget used for feature computation with respect to: (i) the ability of feature values to grasp the problem characteristics, and (ii) the gain in approximation quality when solving an unknown problem instance using automated (feature-informed) algorithm selection.

- We report a comprehensive study using a broad range of $\rho$MNK-landscapes and a portfolio of three variants of the MOEA/D algorithm. We show that different global trends of feature values and non-trivial cost-vs-accuracy trade-off(s) can be distinguished. Depending on the feature under consideration, increasing the walk length or increasing the number of visited neighbors do not always lead to more informative feature values. Besides, controlling the cost of features computation can improve the accuracy of the automated algorithm selection task. Interestingly, this holds independently of whether the sampling cost is taken into account in the available budget, or not.

## 4.2 Sampling Strategies

Feature computations require two time-consuming steps: generating the sample of solution and computing the numerical features from the sample. This second point does not require any calls to the fitness function, since all the solutions used have already been evaluated during the first step. Therefore, the cost of the second step relates only to the real time needed by the machine to make the computation. As this study focuses on black-box complexity, the real time of the machine is ignored and only the number of calls to the fitness function is taken into account. Thus, in the rest of this chapter, only the cost of the sampling step is taken into account. In order to simplify this work, we focused on the most used sampling strategies for landscape analysis: random walks and multi-objective adaptive walks. Despite that this work focus on multi-objective optimization, the approach is compatible with single-objective landscape analysis except for adaptive walks sampling strategies which slightly differs between single-objective and multi-objective optimization. Additional control parameters are introduced to the two sampling methods in order to ensure a greater control on the cost.

### 4.2.1 Random and Adaptive Walks

Two methods are generally used in the literature [55, 21] in order to compute the input walk, namely *random walk* and *adaptive walk*, as introduced previously in Chapter 2. In brief, random walk select randomly a solution in the neighborhood at each iteration. The number of iteration is a user-defined parameter $\ell$. In an adaptive walk, the next solution is selected among the improving neighbors. In a multi-objective setting, improving neighbors are considered with respect to the dominance relation [95, 55]. The walk ends if

no solution improves the current solution. Unlike a random walk, the length of an adaptive walk is determined by the number of steps required to fall into a Pareto local optimal solution, hence the user do not control the length of the walk.

### 4.2.2 Sampling Strategies with Truncated Neighborhoods

We apply simple modifications to the previous mechanisms in order to control more finely the feature cost. We consider two options: (i) to control the walk length $\ell$, and (ii) to control the number of neighbors that need to be evaluated.

The first option is only possible when using a random walk, since the termination step of an adaptive walk cannot be controlled explicitly. This is however a natural option in order to adjust the overall budget, which was not studied systematically. In fact, the length of a random walk is in general set empirically by relying on the end user's expertise. The second option applies for both random and adaptive walks. More specifically, for any integer value $r \leq |\mathcal{N}|$, let us call a truncated neighborhood, denoted $\widetilde{\mathcal{N}}_r$, a neighborhood obtained from the original neighborhood $\mathcal{N}$ by considering solely $r$ neighbors, i.e., $\forall x \in X$, $\widetilde{\mathcal{N}}_r(x) = \{y^1, y^2, \cdots, y^r \mid \forall j \in [1..r], y^j \in \mathcal{N}(x)\}$. As such, we use a truncated neighborhood having $r$ solutions sampled uniformly at random from $\mathcal{N}$. Then, we compute the different features on the basis of the so-defined truncated neighborhood. Algorithms 7 and 8 show the pseudocode of the truncated random walk and the truncated dominance walk, respectively.

On the one hand, these modifications do not affect the sequence of solutions of a random walk, since a random walk only requires picking a neighbor at random. By contrast, for an adaptive walk, at most the $r$ solutions of $\widetilde{\mathcal{N}_r}$ are considered for selecting the (first) dominating neighboring (if any) at each step $t$. This means that an adaptive walk using $\widetilde{\mathcal{N}_r}$ can stop earlier at any step $t$, if the $r$ randomly-selected neighbors do not include any locally dominating solution, and even if there exist some solutions in $\mathcal{N}(x^t) \setminus \widetilde{\mathcal{N}_r}(x^t)$ that dominate $x^t$. In particular, notice that the last solution of the so-obtained adaptive walk is not necessarily a Pareto local optima with respect to the original neighborhood $\mathcal{N}$. On the other hand, independently of the walk type (i.e., random or adaptive), for every solution $x^t$ in the so-computed walk, solely the $r$ solutions of $\widetilde{\mathcal{N}}_r(x^t)$ are evaluated when computing the statistic required for the final feature value. In other words, although we follow exactly the same feature specification as discussed previously, by replacing the (full) neighborhood by its $r$-truncated variant, the so-computed feature values can be different.

Additionally, the previous simple modification reduces the feature budget

to $\Theta(\ell \cdot r)$. However, it is not clear how this impacts the feature values, nor it is clear how much the so-extracted values are still meaningful. Besides, one may wonder whether a specific choice of the budget is to be preferred to accurately characterize the underlying landscapes. In fact, different combinations of $r$ and $\ell$ values could be considered, while still constraining the overall budget to the same pre-fixed value.

**Algorithm 7** Truncated Random walk

**Input:** $l \in \mathbf{N}$: length of the walk.
$r \in [1, \mathbf{N}]$: proportion of explored neighborhood.
$f : \mathcal{X} \longrightarrow \mathcal{Z}$: fitness function.

$x \leftarrow$**random solution**
$S \leftarrow \{(x, f(x))\}$ ▷ evaluate the solution
$c \leftarrow 0$
**repeat**
  $\widetilde{\mathcal{N}}_r(x) \leftarrow$ *Random partition*$(\mathcal{N}(x), r)$
  **for** $x_i \in \widetilde{\mathcal{N}}_r(x)$ **do**
    $S$.append$(\{(x_i, f(x_i))\})$ ▷ evaluate the solution
  **end for**
  $c+ = 1$
  $x \leftarrow$ *Uniform Random*$(\widetilde{\mathcal{N}}_r(x))$
**until** $c = l$
**return** $S$

## 4.3 Modified Accuracy Indicators and Experimental Settings

When taking the sampling cost into account, it is not clear how to adapt indicators to evaluate the machine learning models. Some indicators, such as the $R^2$, only evaluate the learning model ability to make predictions, hence can remain unchanged. The merit measures the gain of using the selected algorithm with respect to using the best algorithm on average. In the following, we propose a modification to include the cost of the sampling into the merit indicator.

**Algorithm 8** Truncated Multi-objective adaptive walk

**Input:** $f : \mathcal{X} \longrightarrow \mathcal{Z}$: fitness function.

$x \leftarrow$**random solution** ▷ set the initial solution
$S \leftarrow \{(x, f(x))\}$ ▷ evaluate the solution
**repeat**
  Dominating $\leftarrow \{\}$
  $\widetilde{\mathcal{N}}_r(x) \leftarrow$ *Random partition*$(\mathcal{N}(x), r)$
  **for** $x_i \in \widetilde{\mathcal{N}}_r(x)$ **do**
    $S$.append$(\{(x_i, f(x_i))\})$ ▷ evaluate the solution
    **if** $x_i$ dominate $x$ **then** *Dominating*.append$(x_i)$
    **end if**
  **end for**
  $x \leftarrow$ *Uniform Random*(Dominating) ▷ select the next solution
**until** $x = NULL$
**return** $S$

### 4.3.1 Cost-integrated Merit Indicator

For performance assessment, we consider an adaptation of the merit measure [59]. We recall that the merit allows to compare the algorithm selected by the machine learning model to the so-called single best solver (SBS) and virtual best solver (VBS).

Let $\mathsf{rhv}(A, i)$ be the average relative hypervolume deviation of a given algorithm $A$ using the maximum allowed budget $\mathsf{B}$ on an instance $i$. Similarly, let $\mathsf{rhvc}(A, i)$ be the average relative hypervolume deviation of the algorithm $A$ when subtracting the cost of feature computation $\mathsf{FB}$, that is when running the algorithm with a budget of $\mathsf{B} - \mathsf{FB}$. Let $\widetilde{\mathsf{rhvc}}(A, J)$ and $\overline{\mathsf{rhv}}(A, J)$ be the average over a given set of instances $J$ of the relative hypervolume deviation, respectively, when the feature cost is or not taken into account. The *single best solver* (SBS) is the algorithm with the best $\overline{\mathsf{rhv}}$ value on the training set $\mathcal{T}$; i.e., $SBS = \arg\min_A \overline{\mathsf{rhv}}(A, \mathcal{T})$. Notice that the SBS does not need any feature computation. The *virtual best solver* (VBS) is the oracle, providing the best $\mathsf{rhv}(\cdot, i, \mathsf{B})$ value for each testing instance $i \in \mathcal{I}$. Let us call the Automated Solver (AS) the algorithm selected by the trained model. The *merit* measures the gain in quality of AS relatively to SBS and VBS. For the purpose of our analysis, we consider the variant of the merit where the sampling cost is taken

into account ($\mathsf{m}'$):

$$\mathsf{m}' = \frac{\widetilde{\mathsf{rhvc}}(\mathrm{AS}, \mathcal{I}) - \overline{\mathsf{rhv}}(\mathrm{VBS}, \mathcal{I})}{\overline{\mathsf{rhv}}(\mathrm{SBS}, \mathcal{I}) - \overline{\mathsf{rhv}}(\mathrm{VBS}, \mathcal{I})}$$

As for the merit introduced in Chapter 3, a cost-integrated merit of $\mathsf{m}' = 0$ corresponds to the perfect oracle and a cost-integrated merit $\mathsf{m}' < 1$ (respectively $\mathsf{m}' > 1$) indicates that using the model performs better (respectively worse) than SBS.

### 4.3.2 Parameter Setting

The $\rho$MNK-landscapes are the same 1 000 bi-objective instances used in Chapters 2 and 3. We consider a truncated neighborhood $\widetilde{\mathcal{N}}_r$ of size $r = \alpha \cdot n$, where the parameter $\alpha$ is in $\{0.05, 0.1, 0.25, 0.5, 1\}$. Notice that $\alpha = 1$ corresponds to the full original neighborhood, whereas for $\alpha = 0.05$ a sample of 5% of the original neighborhood is explored. The two types of walks are experimented. For a random walk, the length is set in the range $\ell \in \{5, 10, 25, 50, 100\}$. For each instance, we thus obtain $1_{(\text{rnd. walk})} \times 5_{(r)} \times 5_{(\ell)} + 1_{(\text{adapt. walk})} \times 5_{(r)} = 30$ possible values for each of the three features set summarized in Appendix 6.2.

## 4.4 Preliminary Experimental Results

In the first part of this study, we compute features from several truncated walks with different parameters. All multi-objective feature sets are used, i.e., dominance-based, indicator-based and decomposition-based features. Before understanding the impact of the sample size in higher-levels tasks, we first focus on two measures: (i) the variation in feature values and (ii) the variation in feature correlations with benchmark parameters.

### 4.4.1 Random Walk Analysis

We start by studying the impact of the different sampling configurations when *a random walk* is considered. We use two measures to elicit the relationship between the setting of $\ell$ and $r$, and the behavior of the so-computed feature values. Firstly, we compute the deviation of feature values with respect to the highest budget-demanding setting of the sampling. More precisely, for every instance, we compute the value $V_{\max}$ for every feature $\mathsf{f}$, obtained when $\alpha = 1$ and $\ell = 100$. Then, we compute for every other value $V_{r,\ell}(\mathsf{f})$ of feature $\mathsf{f}$, obtained with other settings of $r$ and $\ell$, the relative deviation to $V_{\max}(\mathsf{f})$, that is, $|V_{\max}(\mathsf{f}) - V_{r,\ell}(\mathsf{f})|/V_{\max}(\mathsf{f})$. Secondly, for each feature (in each sampling

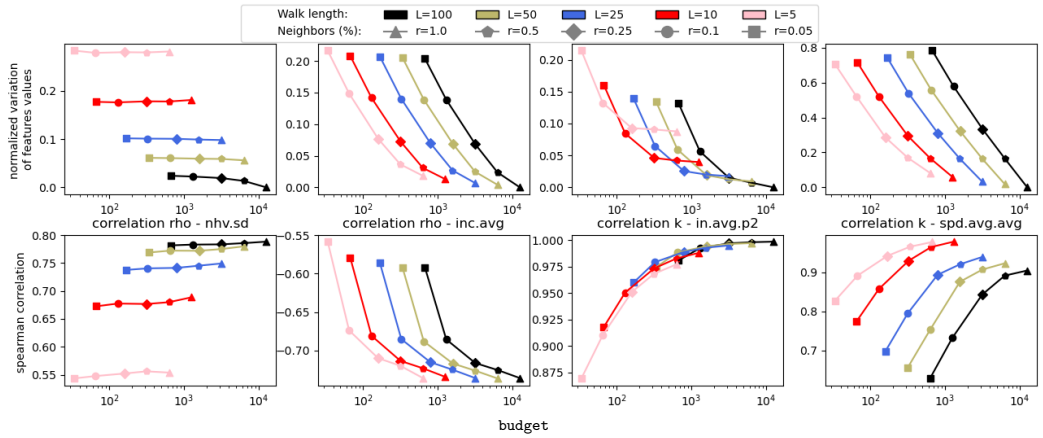


Figure 4.2: Average feature relative deviation (top), and feature correlation (bottom) with $\rho$ (first and second subfigures) and $K$ (third and fourth subfigures) for random walks. From left to right: nhv.sd, inc.avg, in.avg.p2 and spd.avg.avg. The *x-axis* (in log-scale) refers to the number of evaluations of the sampling.

setting), we compute the Spearman correlation between the feature value and the value of $\rho$ and $K$ over the considered $\rho$MNK.

**Preliminaries:** Before going into further details, we found that 30% of features are neither correlated to $\rho$ nor $k$, hence being of limited interest in the following analysis, given the extensive number of considered configurations. They mostly consist of features relating to the first auto-correlation coefficient $\star$.r1 or the kurtosis statistic $\star$.kr. Among the remaining features, we are able to elicit 4 classes where we observe specific trends of feature values and correlation as a function of the setting of $\ell$ and $r$. In the first and second classes, containing respectively 20% and 15% of the overall features, feature values depend exclusively on either $\ell$ or $r$. In the third and fourth classes, containing respectively 15% and 20% of the features, they expose a more complex dependency on the values of $\ell$ and $r$. These four classes are illustrated in Figure 4.2, reporting the average values of the relative deviation over the different instances (top), and the evolution of the Spearman correlation coefficients (bottom), as a function of the sampling cost. In Figure 4.2, two features are taken from the decomposition-based set [21], namely the average maximal distance between improving sub-problems *spd.avg.avg* and the average proportion of improving neighbors for each sub-problem *in.avg.p2*. The two other features are taken from the dominance- and indicator-based set [55], namely

the average number of incomparable solutions *inc.avg* and the standard deviation of the neighborhood hypervolume *nhv.sd*. This is discussed in more details below.

**First and second feature classes:** In the first class, illustrated by feature *nhv.sd* in Figure 4.2 (first column), the feature values are mostly impacted by the length of the walk $\ell$, while being independent to the value of $r$. More precisely, the higher the value of $\ell$, the smaller the relative deviation, independently of $r$. This indicates that the features converge to some fixed values as a function of $\ell$. However, the proportion of evaluated neighbors $r$ has almost *no* impact on the feature values, while implying a substantially higher cost. For instance, computing *nhv.sd* using a 5%-truncated neighborhood and $\ell = 100$ achieves a relative deviation lower than 0.05%, while being 20 times less expensive than a full neighborhood exploration. Besides, our observations with respect to the relative deviation values stay consistent when looking at the Spearman correlations. For instance, the *nhv.sd* feature is mostly correlated with the parameter $\rho$ of the considered landscapes. The correlation changes consistently with respect to the relative feature deviation. In particular, the larger $\ell$, the higher the correlation with $\rho$. However, the neighborhood proportion has almost no impact. Interestingly, exactly the opposite behavior can be reported for other features, which constitute our second class of features represented by *inc.avg* in Figure 4.2 (second column). In this second class, feature values appear to converge to some value as a function of the neighborhood proportion $r$. However, longer walks have a very small impact on the feature values while leading to a substantially higher cost. For instance, computing the *inc.avg* feature using the full neighborhood and a very small random walk of size $\ell = 5$ achieves less than 2.5% deviation compared to the most expensive setting, while requiring 20 times fewer evaluations. We also observe that the evolution of the correlation coefficient with respect to $\rho$ follows the same trend as a function of $\ell$ and $r$.

**Third and fourth feature classes:** In the third class of features, represented by *in.avg.p2* in Figure 4.2 (third column), both $\ell$ and $r$ influences substantially the feature value. Roughly speaking, a larger budget, i.e., larger values of $\ell$ and $r$, leads to lower relative deviations. However, this trend is not linear with $\ell$ and $r$. For instance, for a small neighborhood proportion $\alpha \in \{0.1, 0.05\}$, a clear gap is observed with the other values of $\alpha \in \{0.25, 0.5, 1\}$. A similar observation can be made for relatively small walks of length $\ell \in \{5, 10\}$. However, combining a random walk of length

Table 4.1: Proportion of classes for each subset of features.

| Classes | Features subsets | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | fv | fd | in | spd | inf | sup | inc | hv | hvd | nhv | lnd | lsupp |
| Class 1 (length) | 0.0 | 0.15 | 0.05 | 0.2 | 0.0 | 0.0 | 0.0 | **0.4** | 0.2 | **0.6** | 0.0 | **0.33** |
| Class 2 (neighborhood) | 0.0 | **0.36** | 0.05 | 0.2 | 0.2 | 0.0 | 0.2 | 0.0 | 0.2 | 0.0 | **0.4** | **0.33** |
| Class 3 (linear) | **0.77** | 0.1 | **0.23** | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Class 4 | 0.05 | 0.1 | 0.0 | **0.4** | **0.4** | **0.4** | **0.6** | 0.2 | 0.0 | 0.0 | 0.0 | 0.0 |
| Mixed shape | 0.0 | 0.1 | **0.23** | 0.2 | 0.2 | **0.4** | 0.0 | 0.0 | 0.0 | 0.0 | 0.2 | 0.0 |
| Unrecognizable shape | 0.16 | 0.15 | 0.41 | 0.0 | 0.2 | 0.2 | 0.2 | 0.4 | **0.6** | 0.4 | 0.4 | 0.33 |

$\ell = 25$ and a neighborhood proportion of $r = 25\%$, is enough to achieve a relative deviation lower than 3% while using 16 times fewer evaluations compared against the most expensive setting. Notice that the correlation coefficient increases consistently, as commented previously for the other classes. Finally, in the fourth class of features, represented by spd.avg.avg in Figure 4.2 (fourth column), the relative deviation decreases consistently with the cost of sampling. However, the correlation coefficient does not follow the same trend in the sense that intermediate settings can provide substantially higher correlations. For instance, computing the *spd.avg.avg* feature using a small walk of length $\ell \in \{5, 10\}$ and the full neighborhood shows significantly larger correlation values compared against the most expensive setting, while requiring a significantly lower budget.

**Relation between features, statistical functions and the sample cost:** To improve our understanding of the relation between features and the sample cost, we present in Table 4.1 the proportion of multi-objective features per classes, for each feature subset. Similarly, we investigate the relation between the used statistical function and feature classes in Table 4.2.

As the distribution of features into classes is not an automated process, some difficulties were encountered for several multi-objective features. In Tables 4.1 and 4.2, a distinction was made between features that are difficult to classify manually. While we observe a pattern most of the time, in some cases, it is difficult to decide among which of the four categories the feature belong to. These cases are called *mixed shape* in the Tables 4.1 and 4.2. In particular, it appends when the observed pattern varies between $K$ and $\rho$ or when patterns are not consistent between the two observed measures (correlation or average feature deviation). But in these cases, it is always possible to detect

Table 4.2: Proportion of classes for each statistical function used in the computation of the feature.

| Classes | Statistics functions used in feature computation | | | | | | |
|---|---|---|---|---|---|---|---|
| | .kr | .sk | .r1 | .sd | .avg | .p1 | .p2 |
| Class 1 (length) | 0.0 | 0.22 | 0.11 | 0.21 | 0.04 | 0.07 | 0.15 |
| Class 2 (neighborhood) | 0.0 | 0.22 | 0.0 | 0.13 | **0.34** | 0.0 | 0.15 |
| Class 3 (linear) | 0.0 | 0.0 | 0.0 | **0.26** | 0.17 | **0.38** | **0.38** |
| Class 4 | **0.55** | **0.44** | 0.05 | 0.08 | 0.07 | 0.0 | 0.07 |
| Mixed shape | 0.0 | 0.0 | 0.05 | 0.21 | 0.21 | 0.0 | 0.07 |
| Unrecognizable shape | 0.44 | 0.11 | **0.77** | 0.08 | 0.13 | 0.53 | 0.15 |

the presence of patterns according to the size of the walk and the proportion of the neighborhood explored. Conversely, some features do not belong to any classes (Unrecognizable shape in the Tables). This manual classification is approximate, and we aim to get insight of the relations between the feature, the used statistic and the sample size.

In Table 4.1, each row corresponds to the set of variations for the same feature set, and each column shows the proportion of features in the four classes. Notice that the values have been rounded, which explain why their sum is not exactly 100%. The observed metric has undoubtedly the highest impact on the relation between the multi-objective feature and the sample cost. For example, 77% of features from the fitness value fv are linear (class 3) with the cost of the sample. Similarly, the difference of hypervolume hvd, the proportion of incomparable neighbors inc, and the neighborhood hypervolume nhv have most of their features in a single class. For some metrics, the multi-objective features fall in different classes. For example, the fitness difference fd has at most 36% of features in a single class. In these case, the relation between the length of the walk $\ell$ and the proportion of visited neighbors $r$ and the feature is mostly impacted by statistical functions used in the computation. We observe such results in Table 4.2. For example, the kurtosis kr and the skewness sk are respectively a measure of the *tailedness* of a distribution and of the asymmetries of a distribution. Half of the features computed using either the kurtosis or the skewness end up in the fourth class. A second example is the first auto-correlation coefficient r1. This function was proved to make efficient features [96, 55, 46], however no patterns could be observed in most features.

**Discussion:** From the previous observations, we can say that there is a complex interaction between the feature values and the sampling configuration. In particular, a higher budget does not systematically lead to more consistent values, independently of the considered feature. Moreover, the settings of $\ell$ and $r$ can lead to seemingly different features exposing different trade-offs both in terms of feature values, feature cost, and feature ability to grasp the global characteristics of the underlying landscape. Hopefully, our analysis suggests that there could be some setting of $\ell$ and $r$ leading to reasonably cheap and accurate features. This will be studied in more details in Section 4.5. In the following, we first complement our analysis with respect to the adaptive walk.

### 4.4.2 Adaptive Walk Analysis

**Feature values:** Firstly, we report in Figure 4.3 (left) the relative deviation of features with respect to the values obtained using an adaptive walk with the largest budget ($\ell = 100$ and $r = 100\%$). We observe that the relative deviation spans a wide range (from 2% to more than 60%), as a function of $r$. Actually, the largest deviations are with respect to very small $r$ values. This is to contrast with random walks where, although $r$ was found to have a significant impact, the range of the relative deviation is relatively small. This indicates that the size of the $r$-truncated neighborhood is critical for an adaptive walk. In general, the feature values converge as $r$ increases. However, some features are found to be less sensitive to $r$, as illustrated by the in.avg.p2 and the spd.avg features, for which a small value of $r$ allows to obtain a significantly lower deviation range. Overall, an adaptive walk visiting 25% of the neighborhood provides feature values which are relatively close to a full neighborhood (up to 5%), while requiring a significantly lower budget. One should however keep in mind that given the range of feature values, even such a small deviation is not necessarily negligible.

**Correlation values:** Secondly, in Figure 4.3 (middle and right), we report the evolution of the correlation between feature values and the benchmark parameters $\rho$ and $K$. In the situations where a feature shows a significant (positive or negative) correlation with either $\rho$ (e.g., rhv.sd or inc.avg) or $K$ (e.g., spd.avg) when using a full neighborhood, the strength of the correlation decreases when extracting the feature with decreasing values of $r$. For the other situations, no general tendency can be reported. From our collected data, we can state that computing features with an adaptive walk using the original neighborhood seems to expose the strongest correlation with the benchmark characteristics.

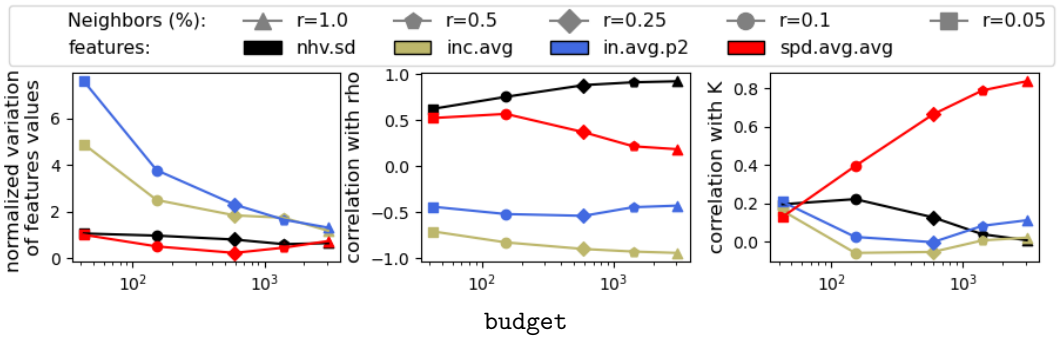


Figure 4.3: Average feature relative deviation (left), and feature correlation with $\rho$ (middle) and $K$ (right) for adaptive walks. The *x-axis* (in log-scale) refers to the number of evaluations of the sampling.

## 4.5 Impact of Cost in High-Level Prediction Tasks

In the previous sections, we studied the impact of the sampling cost/strategy on feature values, as well as, on the correlation of feature value with the landscape global parameters. In this section, we aim at analyzing the impact of using different budgets/samplings when integrating the so-computed features when performing other machine learning tasks, namely, predicting benchmark parameters and selecting the best algorithm for a given problem instance. Let us recall that the higher the feature cost, the smaller the budget of the algorithm.

### 4.5.1 Predicting Benchmarks Parameters

**Task description:** This task informs about the power of a predictive model as a function of the sampling. We train one model with each possible configuration of the sampling in order to fit respectively parameters $K$ and $\rho$ of the input benchmarks. As in Chapter 3, we use two random forest [15] regression model for $\rho$ and $K$. Each model is trained on the basis of the input feature values, using a default value of 100 trees [69]. We then focus on the models' $R^2$ values, as summarized in Figure 4.4. The higher the $R^2$ value, the more explainable the variance in the benchmark parameter ($\rho$ or $K$), using the underlying sampling.

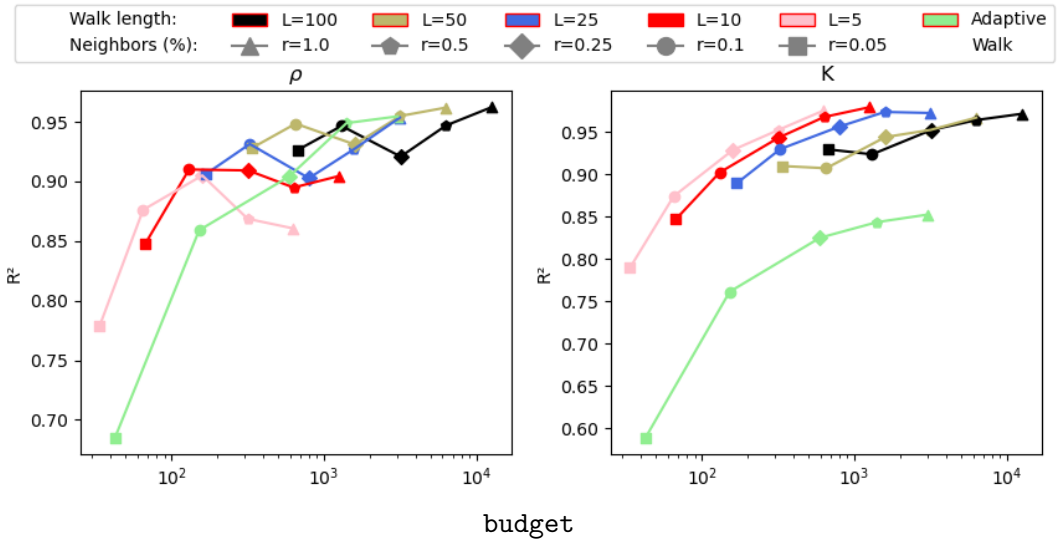


Figure 4.4: Average $R^2$ (over 50 repetitions) of random forest models for predicting $\rho$ (left) and $K$ (right). The *x-axis* (in log-scale) refers to the number of evaluations.

**Results and discussion:** From Figure 4.4, we observe that in comparison to a random walk, an adaptive walk provides a poor trade-off in terms of the $R^2$ value and the implied budget for $k$. We also notice that the $R^2$ has a global tendency to increase as the budget underlying the sampling increases. However, reasonably high $R^2$ values are obtained at lower costs by adjusting the sampling parameters. For instance, an adaptive walk with half of the neighborhood ($r = 50\%$) obtains almost the same $R^2$ values (0.94 for $\rho$ and 0.84 for $K$) while reducing the feature cost by half in average. Looking more carefully at the random walk configurations, we find a number of settings of $\ell$ and $r$ that obtain the same high level of accuracy, but with a drastic saving in the budget when compared against the most expensive setting. Interestingly, we found that the configuration providing the best trade-offs are different depending on the target prediction task. For $K$, a very small random walk ($\ell = 5$) and the full neighborhood exploration ($r = 100\%$) provides among the highest $R^2$ values (0.97) while requiring 20 times less budget than the most expensive setting. In fact, it appears that the most important sampling parameter for predicting $K$ is the proportion $r$ of the neighborhood. For $\rho$, a configuration using a walk length $\ell = 50$ and very small proportion of the neighborhood $r = 10\%$ provides among the highest $R^2$ values (0.94) while requiring 20 times less budget than the most expensive setting. Actually, the $R^2$ value with respect to $\rho$ is mostly impacted by the length of the walk,

Table 4.3: Merit average values without including sample cost for the different sampling configurations.

| | Merit without feature cost ($\mathsf{m}'$) | | | | | |
|---|---|---|---|---|---|---|
| | Random Walk | | | | | Adaptive |
| $r$ | $\ell = 100$ | $\ell = 50$ | $\ell = 25$ | $\ell = 10$ | $\ell = 5$ | Walk |
| 100% | 0.513 | 0.517 | 0.555 | 0.573 | 0.573 | 0.617 |
| 50% | 0.507 | 0.544 | 0.575 | 0.616 | 0.574 | 0.606 |
| 25% | **0.507** | 0.561 | 0.566 | 0.637 | 0.631 | 0.626 |
| 10% | 0.538 | 0.579 | 0.563 | 0.606 | 0.583 | 0.670 |
| 5% | 0.547 | 0.572 | 0.550 | 0.637 | 0.644 | 0.690 |

provided that the proportion of evaluated neighbors is at least $r = 10\%$.

### 4.5.2 Automated Algorithm Selection

**Task description:** The second and more sophisticated task aims at studying the impact of the sampling cost when tackling the automated algorithm selection problem [77] using a feature-informed performance prediction approach. For this purpose, we consider the exact same portfolio of algorithms as in Chapter 3 and the same $\rho$MNK-landscape instances. Following a standard supervised machine learning approach, we split the 1000 $\rho$MNK-landscape instances into a training set $\mathcal{T}$ and a testing set $\mathcal{I}$. We then train a multi-output random forest regression model [15] in order to fit the relative hypervolume deviation of the different algorithms on the basis of the feature extracted with respect to each instance in the training set $\mathcal{T}$. Afterward, given an unseen instance from the testing set $\mathcal{I}$, the features are first computed, and the performance of each algorithm is then predicted on the basis of the so-trained model. The algorithm with the best predicted performance is selected as the recommended one. Since extracting the features has a cost, we distinguish two scenarios for the purpose of our analysis. In the first scenario, the selected algorithm is run with the maximum budget $\mathsf{B}$, hence not counting the sampling cost. In the second, more realistic scenario, the algorithm is run with a budget of $\mathsf{B} - \mathsf{FB}$, where $\mathsf{FB}$ is the number of evaluations used to compute the features. We adopt a standard repeated random hold-out strategy with a 90/10% split for training/testing, and we report the average merit value over 100 independent folds.

Table 4.4: Cost-including merit average values for the different sampling configurations.

| | Merit with feature cost ($\mathsf{m}$) | | | | | |
|---|---|---|---|---|---|---|
| | Random Walk | | | | | Adaptive |
| $r$ | $\ell = 100$ | $\ell = 50$ | $\ell = 25$ | $\ell = 10$ | $\ell = 5$ | Walk |
| 100% | 0.537 | 0.529 | 0.562 | 0.577 | 0.576 | 0.624 |
| 50% | 0.521 | 0.552 | 0.580 | 0.620 | 0.577 | 0.610 |
| 25% | **0.515** | 0.566 | 0.569 | 0.640 | 0.634 | 0.629 |
| 10% | 0.544 | 0.583 | 0.566 | 0.609 | 0.585 | 0.672 |
| 5% | 0.551 | 0.575 | 0.552 | 0.639 | 0.646 | 0.692 |

**Results and Discussion:** In Tables 4.3 and 4.4, we report the merit values obtained for the different configurations. First, we observe that the merit value $\mathsf{m}$, taking into account the feature cost, is always worse than the merit without cost $\mathsf{m}'$. With no surprise, this indicates that reducing the budget devoted to feature computation can help to improve the overall performance. More importantly, the merit $\mathsf{m}$ is always lower than 1. This means that a feature-informed automated algorithm selection approach is always better than SBS, independently of the experimented sampling configuration. This is especially interesting for the cheapest configuration ($\ell = 5, r = 5\%$), implying an extremely constrained feature budget of at most 50 evaluations. Besides, a random walk provides better merit values than an adaptive walk. The merit increases when the sampling size decreases. More precisely, the loss of accuracy is slightly more important when the length of random walk is reduced. We also find that using the most expensive sampling configuration ($\ell = 100, r = 100\%$) is not the most efficient in terms of merit. In fact, a random walk with $\ell = 100$ and $r = 25\%$ requires four time less budget while providing the best merit values. Surprisingly, this holds independently of whether the cost of sampling is taken into account or not. This gain can be attributed to the fact that a relatively small budget (i.e., 0.5% of the global budget) still allows computing informative features leading to accurate predictions, while leaving more chance for the algorithm to converge. This is consistent with the results from our first task, where we were able to elicit some cost-vs-accuracy trade-offs when varying the sampling parameters $\ell$ and $r$ for predicting the values of $K$ and $\rho$.

Finally, we use the mean Gini impurity measure to extract the importance of each feature from the trained random forest models. We then compute the importance rank of each feature normalized in $[0, 1]$, with 1 being the most important feature. In Figure 4.5, we report the 10 most important features

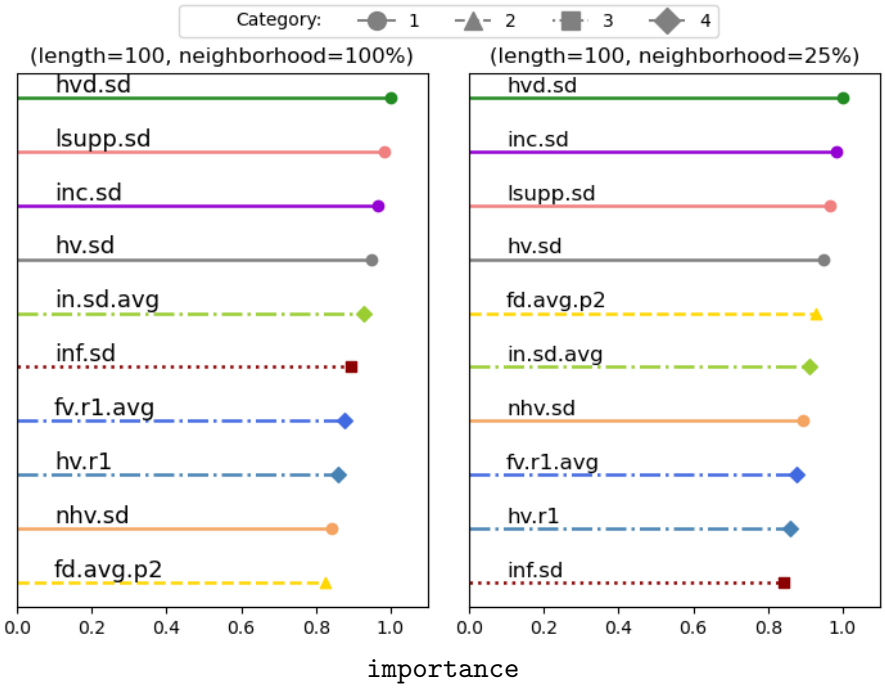


Figure 4.5: Top 10 most important features. Line styles and shapes refer to the feature categories.

when using the most expensive setting ($\ell = 100$, $r = 100\%$, left) and the sampling providing the best merit ($\ell = 100$, $r = 25\%$, right). We clearly see that the ten most important features are the same in both settings. This indicates that using a restricted budget does not necessary lead to a major change in feature importance. However, we can observe small variations in the respective ranks of features. Besides, we found that five features fall into the first category elicited in the analysis of Section 4.4. In particular, the values of the four most important features hvd.sd, lsupp.sd, inc.sd, hv.sd are highly impacted by the length of the walk $\ell$. One feature fd.avg.p2 falls into the second category where the proportion of neighbors $r$ is found to have the greatest impact on this feature value. Another feature inf.sd belongs to the third category, increasing its accuracy according to the sampling size. Finally, three features, namely hv.r1, in.sd.avg, and fv.r1.avg fall into the fourth category, where the highest correlation with the landscape characteristics is observed with a reduced sampling size.

## 4.6 Conclusion

In this chapter, we conducted the first systematic analysis on the impact of sampling on the extraction of multi-objective combinatorial landscape features, and their integration into feature-informed performance prediction and algorithm selection approaches. By adding a parameter to adjust the proportion of the explored neighbors in random and adaptive walks, we investigated the impact of the sample size on features computation. We provide evidence that feature values, and their correlation with benchmark characteristics, expose complex dependencies with the sampling type and cost. In particular, we showed that a random walk using a reasonably small proportion of neighbors leads to cheap and informative feature values. Since a reduction in the feature cost allows the selected algorithm to increase its budget, using features computed with a cheaper cost is beneficial to the automated algorithm selection task. Different questions are however left open. For instance, it would be interesting to extend our analysis to other algorithms, benchmarks, combinatorial domains and objective space dimensions. Following this line, the next chapter proposes a new multi-objective combinatorial benchmark extending MNK-landscapes. The proposed benchmark introduces heterogeneity by setting different levels of difficulty for each objectives.

# Chapter 5

# Heterogeneous Multi-objective NK-Landscapes

Unlike the previous chapters, in this chapter, we consider the design and analysis of landscape features in the context of a *heterogeneous* multi-objective problem. Heterogeneity in multi-objective optimization can be viewed from different perspectives. We are interested in the heterogeneity arising in terms of ruggedness and multi-modality. By adapting the multi-objective NK model, a heterogeneous problem generator is defined and carefully studied through global and local metrics. The new benchmark is thoroughly analyzed with global features for small instances and local features for large instances. Besides, we use the decomposition paradigm to understand the difficulties introduced by the heterogeneous MNK model. Finally, we consider two algorithms, namely MOEA/D and NSGA-II, to observe how heterogeneity impacts the performance of evolutionary algorithms.

The contributions of this chapter were initially published in the Genetic and Evolutionary Computation Conference (GECCO 2022) [23], which was nominated for the best paper award. An extended version was submitted for publication in the European Journal of Operational Research (EJOR). At the time of writing, it is under revision.

## 5.1 Motivations

We consider heterogeneous multi-objective problems where the objective functions differ in one or several aspects, such as scaling, landscape, evaluation time, or theoretical and practical difficulty [35]. Studies on multi-objective evolutionary algorithms do not frequently pay attention to the heterogeneity of the objectives, and to the way it might influence search difficulty. This is particularly the case when considering multi-objective benchmarks, where the focus is usually the global characteristics of the multi-objective landscape, or the shape of the Pareto front. However, real-world problems might exhibit a significant variability in the objectives' characteristics, and recent research has addressed the question of heterogeneous objectives and proposed multi-objective approaches to deal with them [7]. Much of this previous research has been, however, focused on problems where the heterogeneity arises in evaluation times or latencies, that is, when each objective takes a different amount of time to be evaluated [6, 5, 14, 19].

In this chapter, we are interested in problems where the multi-modality and the ruggedness significantly differ among the objectives. As a representative class of problems, we select multi-objective NK-landscapes [1, 3, 53, 95], and we build on recent results analyzing and characterizing this problem class [21, 24, 56, 55]. The parameter $K$ of the NK model critically influences the characteristics of the fitness landscape. As $K$ increases, the ruggedness and the multi-modality of the landscape also increase. Therefore, we investigate multi-objective NK-landscapes where each objective has a different number of variable interactions $K$. Such a model serves as a framework to investigate the impact of heterogeneous objectives, by allowing us to evaluate how the difference in the amount of ruggedness among the NK models translates into the difficulty of the multi-objective problem. Although our analysis could be extended to an arbitrary number of objectives, as in the previous chapters, we focus on bi-objective NK-landscapes. We conduct exhaustive experiments on heterogeneous bi-objective NK-landscapes with different number of variable interactions per objective, and different degrees of heterogeneity. Using traditional metrics from landscape analysis, we analyze the heterogeneous landscapes in terms of multi-modality and of the Pareto set structure. We further analyze to what extent these metrics are impacted by the number of variable interactions and by the degree of heterogeneity among the objectives. Firstly, we introduce two decomposition techniques to better render the impact of heterogeneity on the landscape, inspired by decomposition-based features proposed in Chapter 2. The proposed techniques are global features based on the full enumeration of the search space, and on the mapping of multi-objective

local optimal solutions into the objective space to inform about their distribution.

Secondly, for problems with large number of variables, we investigate the ability of existing multi-objective landscape features in capturing the degree of heterogeneity. We consider the three sets of features investigated in Chapter 2, 3 and 4, namely dominance, indicator, and decomposition-based features. We find that decomposition-based features are particularly well suited to capture heterogeneity. More importantly, this feature-based analysis allows us to provide more insight into the impact of heterogeneity. On the large problem instances, we study the behavior of two state-of-the-art multi-objective evolutionary algorithms, namely MOEA/D [104] and NSGA-II [26], when facing a range of large problems with different degrees of heterogeneity. While MOEA/D is found to perform better regardless of the degree of heterogeneity, the performances of both algorithms are found to be impacted by the degree of heterogeneity similarly.

## 5.2 Problem Definition

We introduce heterogeneous multi-objective NK-landscapes, an extension of MNK-landscapes introduced in Chapter 1, by defining each objective function $F_i$, $i \in \{1, \dots m\}$, as follows:

$$F_i(x) = \frac{1}{n} \cdot \sum_{j=1}^{n} f_j^i(x_j, \Pi_{K_i}(x_j)). \tag{5.1}$$

where the only change is in the definition of the interaction variables $\Pi_{K_i}(x_j)$. It now denotes, for each decision variable $x_j$, a set of $K_i$, $i \in \{1, \dots M\}$, other interacting variables within sub-function $f_j^i$ with respect to the $i$-th objective. In other words, a heterogeneous NK-landscape has now $M$ possibly different numbers of interactions $(K_1, K_2, \dots, K_M)$ respectively to the $M$ objective functions. Since higher values for parameter $K_i$ mean higher non-linearity (among variables), it is well-known that the single-objective landscape with higher values of $K_i$ are relatively more rugged, more multi-modal, and hence more difficult to search [33, 66]. Consequently, having different $K_i$ values for the different objectives allows one to model heterogeneous multi-objective landscapes with a variable difficulty among the objectives.

Notice that there might be different ways to define the sets $\Pi_{K_i}(x_j)$, $j \in \{1, \dots, n\}$ with respect to each objective $F_i$, $i \in \{1, \dots, m\}$. To reduce the variability of the variable interaction structure, the single-objective NK-landscapes

(i.e., the $F_i$ functions) underlying a given heterogeneous MNK-landscape are built using a particular reduction process. More precisely, assume without loss of generality that the $K_i$ values are sorted in a non-decreasing order, i.e., $K_1 \leq K_2 \leq \ldots \leq K_M$. Then, an initial single-objective NK-landscape instance is first generated for the last objective $F_M$ with the highest ruggedness $K_M$ using a standard procedure [9]. This means the set of interaction variables $\Pi_{K_M}(x_j)$ are first generated and fixed, as well as the output tables of the contribution sub-functions $f_j^M$. NK-landscapes instances with lower $K$ are then constructed by using a process called NK-model reduction [23]. For each variable $x_j$, the interacting variable among $\Pi_{K_M}(x_j)$ with the smallest mean squared distance between each possible configuration of the other variables is selected to be removed, that is:

$$x_j = \arg \min_{x_j \in \Pi(x_i)} \sum_{\bar{x} \in x_i \cup \Pi(x_i)/x_j} \left(f_i(x_i, \bar{x} \cup x_j = 1) - f_i(x_i, \bar{x} \cup x_j = 0)\right)^2 \quad (5.2)$$

where $\bar{x} \in X_i \cup \Pi(X_i)/X_j$ represents a configuration of the interactions of $i$ excluding $j$, and $f_i(\bar{x} \in X_i \cup \Pi(X_i)/X_j)$ is the value of function $f_i$ for a particular configuration of the interacting variables. A new sub-function $f_j$ with $2^{M-1}$ values is then randomly generated for each variable and their set of interactions $\Pi_{k_{M-1}}(x_j)$ to build a new NK-landscape defining objective function $F_{M-1}$. The process repeat to construct progressively all instances. This process guarantees that, for any pair of NK models $(NK_1, NK_2)$ generated from a common initial instance and such that $K_1 < K_2$, the $\Pi_{NK_1}(X_j)$ variable interactions of a variable $X_j$ are a subset of $\Pi_{NK_2}(X_j)$.

Considering bi-objective MNK-landscapes, parameters $K_1$ and $K_2$ configure the heterogeneity of the instance. For homogeneous instances, we set $K_1 = K_2$. We define the *total degree of variable interaction* as $T = K_1 + K_2$, and the *degree of heterogeneity* as $D = |K_2 - K_1|$. It is important to notice that, given $T$ and $D$, the sum $T + D = \max(K_1, K_2) \cdot 2$ must be even in order to ensure feasibility, resulting in only two possible symmetrical pairs $(K_1, K_2)$. Fixing $K_1 < K_2$ sets the number of possible configurations to one, therefore the total degree of variable interactions $T$ and the degree of heterogeneity $D$ are two alternative parameters to characterize MNK-landscapes.

The two parameters $T$ and $D$ are actually important to define a diverse set of benchmark instances where the impact of heterogeneity can be studied from a number of orthogonal perspectives. In the following section, and similarly to previous chapters, we restrict our analysis to heterogeneous MNK-landscapes with two objectives ($M = 2$) and additionally, we consider the following scenarios:

- Fixed total degree of variable interactions ($T$),
- Fixed degree of heterogeneity ($D$),
- Increasing heterogeneity without fixing $T$ nor $D$.

## 5.3 Landscape Analysis for Small Heterogeneous MNK-landscapes

In this section, we investigate small instances of heterogeneous multi-objective NK-landscapes. We consider an instance to be *small* when the number of variables is small and that the search space can be fully enumerated. Such characteristic allows us to compute exact statistics on the landscape. In particular, we can compute all Pareto local optima solutions (PLO) and all single-objective local optima solutions (SLO) by enumerating the search space. As a reminder, a solution is a Pareto local optima if its neighbors do not dominate it, and a solution $x \in \mathcal{X}$ is a single-objective local optima according to the objective $F_i$ if, for all solution in its neighbors $x' \in \mathcal{N}(x)$, $x$ has a better objective value $F_i(x) > F_i(x')$. The presence of Pareto local optima solutions and single-objective local optima solutions are known for making it difficult for algorithms to solve an instance.

We considered problems with $N = 14$ and $K \in \{1, \ldots, 12\}$, such that instances are generated by reducing an instance with $K = 12$. For each $K$ value, we generate 50 random instances, and apply the reduction for each of them, obtaining $50 \times 12 = 600$ (single-objective) NK models. Finally, in order to create bi-objective MNK-landscapes, we pair each possible combination of $K$ values for each of the original instances, ending up with 50 instances of the $\frac{12 \times 11}{2} = 66$ bi-objective problems.

### 5.3.1 Visual Inspection of the Objective Space

We start by analyzing the impact of heterogeneity on the structure of the objective space. Given a fixed value of $T = K_1 + K_2 = 13$, we show in Figure 5.1 three exemplary heterogeneous MNK-landscapes in which the level of heterogeneity varies, from slightly heterogeneous objectives (left) to highly heterogeneous objectives (right). First, we observe that the distribution of objective values is different for different levels of heterogeneity. For nearly homogeneous objectives (left), the ellipse surrounding the objective space looks like a circle, where the range of objective values is the same for both objectives,

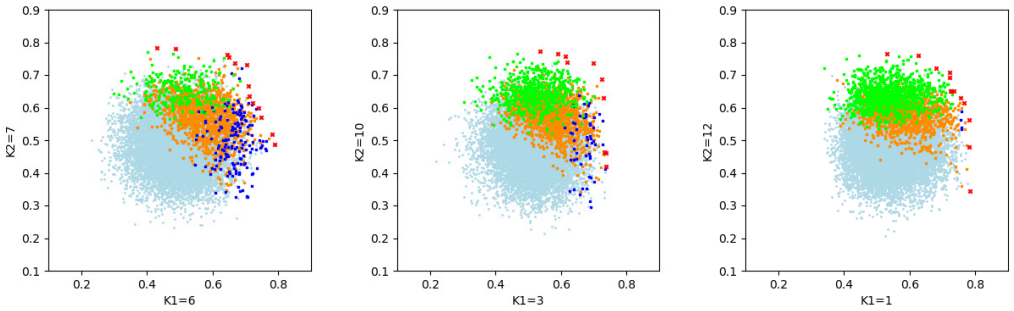


Figure 5.1: Objective space for three heterogeneous MNK-landscapes with a fixed total degree of variable interaction $T = K_1 + K_2 = 13$. The degree of heterogeneity increases from left to right. Red points are Pareto optimal solutions, blue points are single-objective local optima solutions w.r.t. the first objective (which are not Pareto optimal), green points are single-objective local optima solutions w.r.t. the second objective, orange points are Pareto local optima solutions (that are not Pareto optimal, nor single-objective local optima solutions for any objective), and gray points are non-local optimal solutions.

similar to what we observe for homogeneous MNK-landscapes [95]. However, as the degree of heterogeneity increases, the position of solutions in the objective space seems to squeeze for the objective with a smaller $K$. We attribute this to the underlying distribution of objective values for NK models, for which the range is known to increase with K.

Figure 5.1 additionally highlights Pareto optimal solutions, single-objective local optima solutions for each objective and remaining Pareto local optima solutions in different colors. We observe that the PF size and the number of Pareto local optima solutions do not seem to significantly fluctuate with the level of heterogeneity. However, the more homogeneous the objectives, the better distributed the single-objective local optima solutions among both objectives (left). By contrast, for highly heterogeneous objectives (right), almost all local optima are with respect to the second objective ($f_2$), resulting in a proportional increase of Pareto local optima solutions on top of the objective space.

**Pareto Local Optimal Solutions density:** Before going into further details, we show in Figure 5.2 the mapping of solutions in the objective space, as well as the density of Pareto local optima solutions, for the considered heterogeneous MNK-landscapes. The objective-values of all solutions are normalized

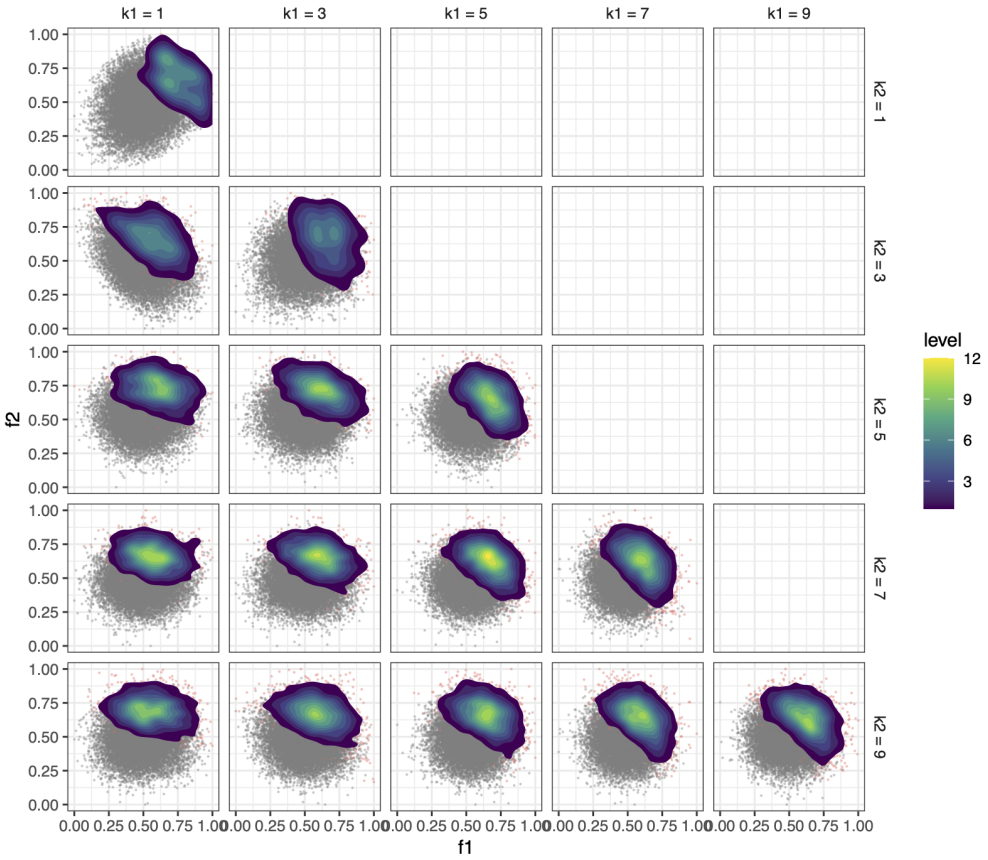


Figure 5.2: Pareto local optima density in the objective space for HMNK-landscapes with $n = 14$. Red points are Pareto local optimal solutions and gray points are non-local optimal solutions. Density levels are computed with a two-dimensional kernel density estimation [98].

according to the minimal and the maximal value of the objective, as introduced in Chapter 1. For each objective $i \in \{1, \ldots, m\}$, the normalized $\hat{F}_i$-value of a solution $x$ is given by:

$$\hat{F}_i(x) = \frac{F_i(x) - \min_{\forall x \in \mathcal{X}} F_i(x)}{\max_{\forall x \in \mathcal{X}} F_i(x) - \min_{\forall x \in \mathcal{X}} F_i(x)} \tag{5.3}$$

Pareto local optima solutions play an important role in the understanding of the multi-objective landscape structure [55]. This is because the difficulty of searching in a given landscape is typically a function of the number and distribution of Pareto local optima solutions. When increasing the heterogeneity

among the objectives, a majority of Pareto local optima solutions tend to be single-objective local optima with respect to the hardest objective. In the reminder of this section, we provide a more focused analysis on the distribution of Pareto local optima solutions. In standard (homogeneous) MNK-landscapes, the density levels of Pareto local optima solutions in the objective space are known to be equally distributed among both objectives [95]. It appears in Figure 5.2 on problems in the diagonal, by a blue filled-in area which expands from the top-center of the objective space to the right-center as the number of interactions is the same for each objective. When the degree of heterogeneity increases, we observe that the density level rotates to the upper part of the objective space. Rugged landscapes have a higher number of single-objective local optima. All local optima are also a Pareto local optima, except in few cases when two local optima are equivalent where only one of the two local optima is a Pareto local optima. Thus, a variation in heterogeneity alters the landscape by shifting the amount and distribution of Pareto local optima solutions towards the most difficult objective.

### 5.3.2 Global Features

**Experimental Settings:** We computed global features on each of the bi-objective problems instances and report the average values among the 50 instances. The metrics are summarized in Table 5.1.

Table 5.1: Global landscape characteristics extracted from MNK-landscapes.

| name | description | |
|---|---|---|
| PF_size | proportion of solutions in the Pareto front | [52, 3] |
| supp | proportion of supported solutions in $R_1$ | [52] |
| hv | hypervolume covered by solution from $R_1$ | [3] |
| cc | proportion of connected components in $R_1$ | [68] |
| sing | proportion of isolated solutions (singletons) in $R_1$ | [68] |
| lcc | prop. size of the largest connected component (lcc) in $R_1$ | [95] |
| lcc_hv | proportion of hypervolume covered by the lcc in $R_1$ | [57] |
| plo | proportion of Pareto local optimal solutions (PLO) | [67] |
| mean_slo | prop. single-objective local optima (SLO) per objective | [57] |
| slo_i | proportion of SLO for objective $f_i$ | |
| p_slo_i | same as above, proportional to the number of PLO | |
| plo_not_slo | proportion of PLO that are *not* SLO for any objective | |
| p_plo_not_slo | same as above, proportional to the number of PLO | |
| slo_dev | deviation of the number of SLO per objective: $\mid$ slo_1 $-$ slo_2 $\mid$ | |
| p_slo_dev | same as above, proportional to the number of PLO | |

**Structure of the Pareto set:** Figure 6.2 (top right and center) summarizes some metrics related to the structure of the Pareto set for all considered heterogeneous bi-objective problems: the number of solutions in the Pareto front, and the number of connected components (`cc`) in the graph $G = (PF, E)$ where there is an edge $e = (x, y) \in E$ if solutions $x$ and $y$ are neighbors. The analysis reveals that the number of Pareto optimal solutions does not vary significantly among the considered problems. For a fixed $K_1$, it seems to slightly decrease with $K_2 > K_1$, which confirms results from homogeneous MNK-landscapes [3, 95]. However, this observation does not hold when fixing $K_2$ and varying $K_1 < K_2$. This suggests that the "most-difficult" objective has more impact on the cardinality of the Pareto set than the "easier" objective for MNK-landscapes. We also observe that there are slightly fewer Pareto optimal solutions as the degree of heterogeneity among the objectives increases. Similarly, the proportion of supported solutions (`supp`) and the global hypervolume (`hv`) reported in Appendix 6.2 do not reveal any significant variations. They mostly relate to the shape of the PF, which does not reveal significant changes in the benchmark. We can however notice that `hv` seems to increase with the number of variable interactions, as for homogeneous MNK-landscapes [3].

In terms of connectedness of the Pareto set, it is reported to decrease with the number of variable interactions for homogeneous MNK-landscapes in [95]. This is what we observe in Figure 6.2, where the number of connected components (`cc`) increases with $K_1$ and $K_2$. We observe that related metrics in Appendix 6.2, such as the proportion of isolated Pareto-optimal solutions (sing), and the size and hypervolume of the largest connected component (lcc_hv), vary accordingly. Interestingly, we do not observe any significant change with respect to the difference between $K_1$ and $K_2$, suggesting that the connectedness is *not* impacted by the level of heterogeneity among the objectives.

**Multi-modality:** Let us now analyze how the non-linearity of objectives, and their level of heterogeneity, influences the multi-modality of the landscape. In Figure 6.2 (top left, and bottom), we report additional metric values related to the number of local optimal solutions for the 50 folds of each pair of $(K_1, K_2)$ values. We focus on a subset of metrics.

For single-objective NK-landscapes, it is expected that higher $K$ values result in more local optima [46]. For homogeneous MNK-landscapes, the number of Pareto local optima solutions is also known to increase with $K$ [95]. Overall, all metrics related to the multi-modality show a regular variation according to both $K$ values: the larger $K$, the more the landscape is rugged. The number of Pareto local optima solutions (`plo`, top right) varies significantly and, as

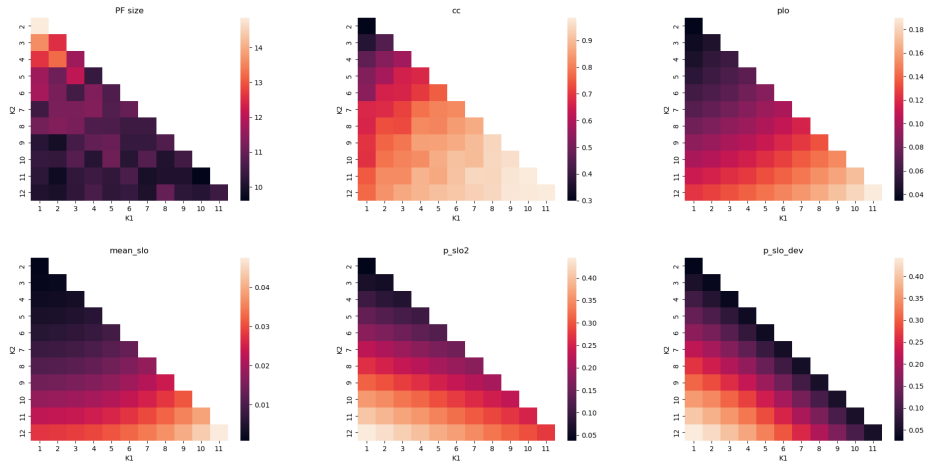


Figure 5.3: Statistics computed from heterogeneous MNK-landscapes.

expected, it clearly depends on the value of $K$ for both objectives. However, the number of Pareto local optima solutions remains constant whatever the level of heterogeneity among the objectives. Similarly, the mean proportion of single-objective local optima solutions per objective (`mean_slo`, bottom left) significantly increases along $K_1$ and $K_2$. Our results reveal that the variation can be significant. As already illustrated in Figure 5.1, the number of single-objective local optima for objective $F_i$ ($\texttt{slo}_i$) increases with $K_i$, and so does its proportion among Pareto local optima solutions. Interestingly, we also observe that the latter increases a bit more significantly as objectives get more heterogeneous, that is when there is a large difference between $K_1$ and $K_2$ (see, e.g., `p_slo_2`, bottom center).

The metrics discussed above, such as `plo` or `mean_slo` but also `cc`, mostly capture this general characteristic of the landscape. Similar to the number of Pareto local optima solutions, the number of Pareto local optima solutions that are not single-objective local optima solutions for any objective `plo_not_slo` increases with the total degree of interactions $T$. However, its normalized variant `p_plo_not_slo` decreases with $T$. This can be interpreted as follows: for smoother landscapes with $(K_1 = 1, K_2 = 2)$, 5% of Pareto local optima solutions are also a single-objective local optima solution with respect to one objective, whereas for highly rugged landscapes with $(K_1 = 11, K_2 = 12)$, more than 90% of Pareto local optima solutions come from single-objective local optima solutions with respect to $F_1$ or $F_2$.

The total degree of variable interactions $T$ captures the general trend of the ruggedness, but fails to grasp the disparity among the objectives. For a low

degree of heterogeneity, we already observed in Figure 5.1 that single-objective local optima solutions are well-balanced between the objectives. To go further, we compute the deviation between the number of single-objective local optima solutions for each objective `slo_dev`, together with its normalized variant, as a proportion among Pareto local optima solutions, `p_slo_dev`. The latter is reported in Figure 6.2 (bottom right) for different $K_1$ and $K_2$ values. Increasing the heterogeneity $D$, such as $K_1 < K_2$, results in a strong concentration of single-objective local optima solutions on the second objective. In other words, slo_dev increases with the objectives' heterogeneity. Contrary to the proportion of Pareto local optima which are not single-objective local optima solutions, that decreases with the total degree of interactions $T = K_1 + K_2$, the proportional variant p_slo_dev is mostly impacted by the degree of heterogeneity $D = |K_1 - K_2|$.

**Correlation Analysis:** To push our analysis further, we report in Figure 5.4 the correlation between the 17 considered metrics and: (1) the number of variable interactions for the first objective $K_1$, (2) the number of variable interactions for the second objective $K_2$, (3) the total number of variable interactions for both objectives $T = K_1 + K_2$, and (4) the degree of heterogeneity among the objectives $D = |K_1 - K_2|$. The correlations reveal five categories of observations. Firstly, as expected, the number of single-objective local optima solutions (`slo_i`) and their proportion within Pareto local optima (`p_slo_i`) are highly correlated with $K_i$, forming two small clusters in the center. The latter is normalized by the number of Pareto local optima solutions, which results in an anti-correlated relationship between `p_slo_i` and the cluster with $K$ of the other objective. The number of Pareto local optima (`plo`), the average number of single-objective local optima solutions per objective (`mean_slo`) and the number of Pareto local optima which are not single-objective local optima solutions (`plo_not_slo`) are all strongly correlated with $T = K_1 + K_2$. So are the proportion of isolated solutions (`sing`) and the proportion of connected components (cc) among Pareto optimal solutions, forming all together a large cluster in the bottom right of the figure. This cluster is related to both previous clusters, where $K_1$ and $K_2$ appear. Inversely, the proportion of Pareto local optima which are not single-objective local optima solutions (p_plo_not_slo) decreases with $T = K_1 + K_2$. Similarly, the size of the largest connected component (`lcc`) follows an opposite trend to the number of isolated solutions (`sing`) and the proportion of connected component (`cc`). In a fourth cluster, we observe that the deviation of single-objective local optima solutions (`slo_dev`) and its proportion (p_slo_dev) are strongly impacted by the

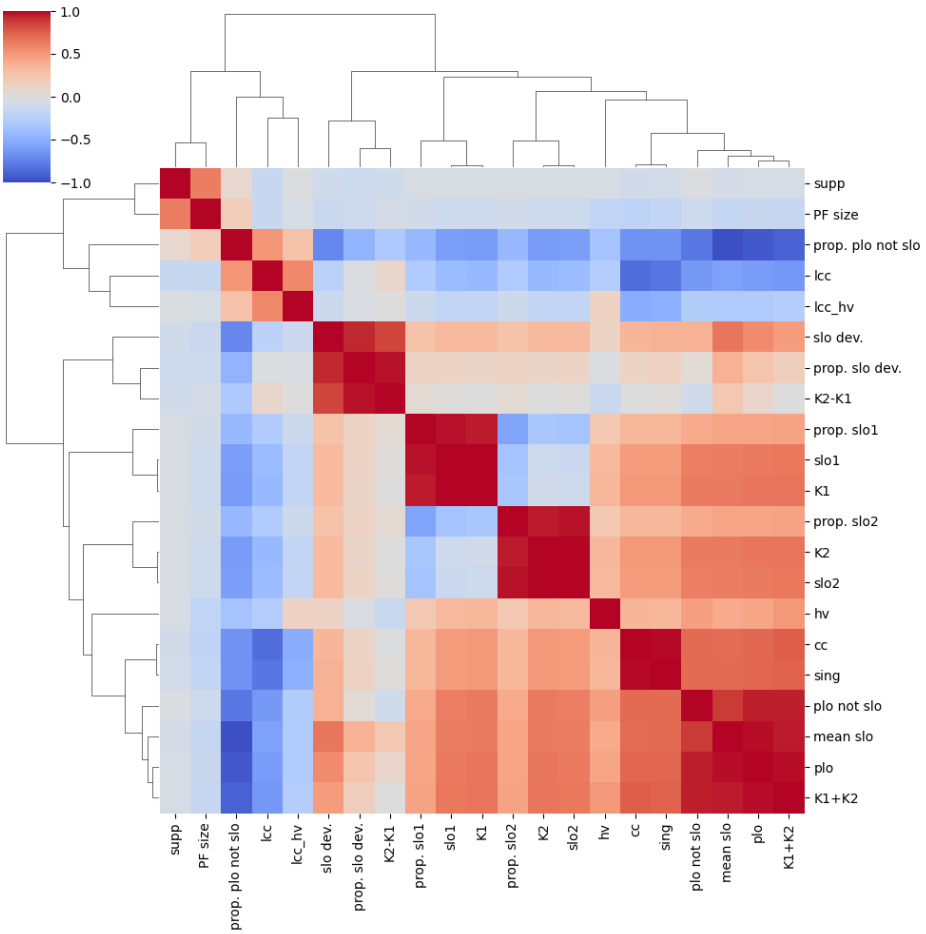


Figure 5.4: Spearman correlation between landscape metrics, $K_1$, $K_2$, the total degree of interactions $K_1 + K_2$, and the degree of heterogeneity $|K_1 - K_2|$.

degree of heterogeneity $D = |K_1 - K_2|$, when slo_dev also reveals a medium correlation with other benchmark parameters ($K_1$, $K_2$, and $T = K_1 + K_2$). As such, for the same degree of heterogeneity, the range of the deviation of single-objective local optima solutions for each objective is impacted by the total degree of variable interactions. Finally, as previously mentioned, the proportion of Pareto-optimal solutions and of supported solutions (`supp`), together with the hypervolume of the largest connected component (`lcc_hv`) do not change much with benchmark parameters.

## 5.4 Distribution of Local Optima

To better analyze the structure of local optimal solutions and their mapping into the objective space, we propose two different strategies based on the decomposition paradigm. More specifically, we adopt an *angle*-based approach, as well as a *scalar*-based approach. It is worth noticing that angular techniques are, like the decomposition paradigm, at the heart of a number of multi-objective evolutionary algorithms, such as RVEA [18]. We here get our inspiration from such techniques, but with the purpose of designing new tools informing about the structure and properties of multi-objective landscapes.

**Distribution based on Angular Decomposition:** Our first approach consists in clustering the solutions on the basis of the angle that their objective-values induce in the objective space. For this purpose, let us consider a set of $\mu$ uniformly-distributed weight vectors $w^i$, $i \in \{1, \ldots, \mu\}$. For each solution $x \in \mathcal{X}$, the angle $\Theta_i(x)$ between the objective vector $F(x)$ and a weight vector $w^i$ is defined by:

$$\Theta_i(x) = \cos^{-1}\Big(\frac{|F(x) \odot w^i|}{||F(x)|| \cdot ||w^i||}\Big) \tag{5.4}$$

where $||\cdot||$ is the usual Euclidean norm and $\odot$ is the scalar product. We then propose to assign each solution to the weight vector minimizing the so-defined angle, i.e., a solution $x$ is assigned to the weight vector $w_i$ such that $\Theta_i(x)$ is minimum, breaking ties arbitrary (see Figure 5.5 for an illustration). Let us denote by $\mathcal{S}_i$ the set of solutions assigned to the weight vector $w_i$, i.e., $\mathcal{S}_i = \{x \mid i = \arg\min_j \Theta_j(x)\}$, and let $\mathcal{S}_i^\star$ be the subset of $\mathcal{S}_i$ containing only Pareto local optima solutions. Intuitively speaking, $\mathcal{S}_i^\star$, $i \in \{1, \ldots, \mu\}$, defines a partition of Pareto local optima solutions with respect to their angular distance in the objective space, i.e., Pareto local optima solutions in the same subset lay in the same region of the objective space defined with respect to the angle of the corresponding weight vector. In Figure 5.6, we then report the distribution of $|\mathcal{S}_i^\star|$, i.e., the number of Pareto local optima solutions in each angular region, as a function of the weight vectors, which allows us to analyze the distribution of Pareto local optima solutions in the objective space.

In the left part of Figure 5.6, we report the average number of Pareto local optima solutions over a set of heterogeneous instances having a fixed number of variable interaction $K_1 = 1$ for the first objective, while varying the number of interactions $K_2$ in the second (more difficult) objective. The distribution of Pareto local optima solutions appears to have a normal shape, with most Pareto local optima solutions being at the center of the objective space, i.e.,

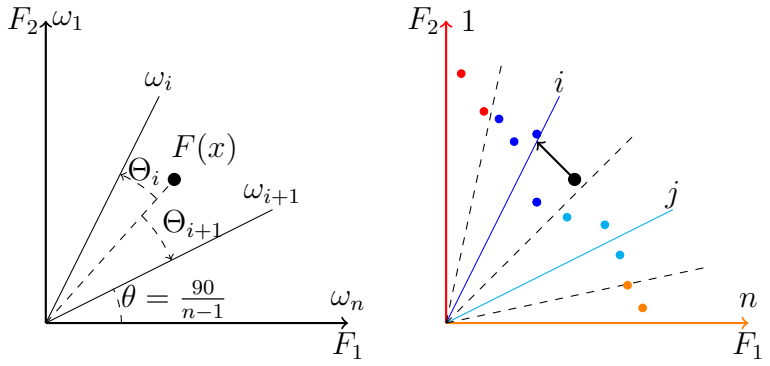


Figure 5.5: Illustration of the angular decomposition of the search space

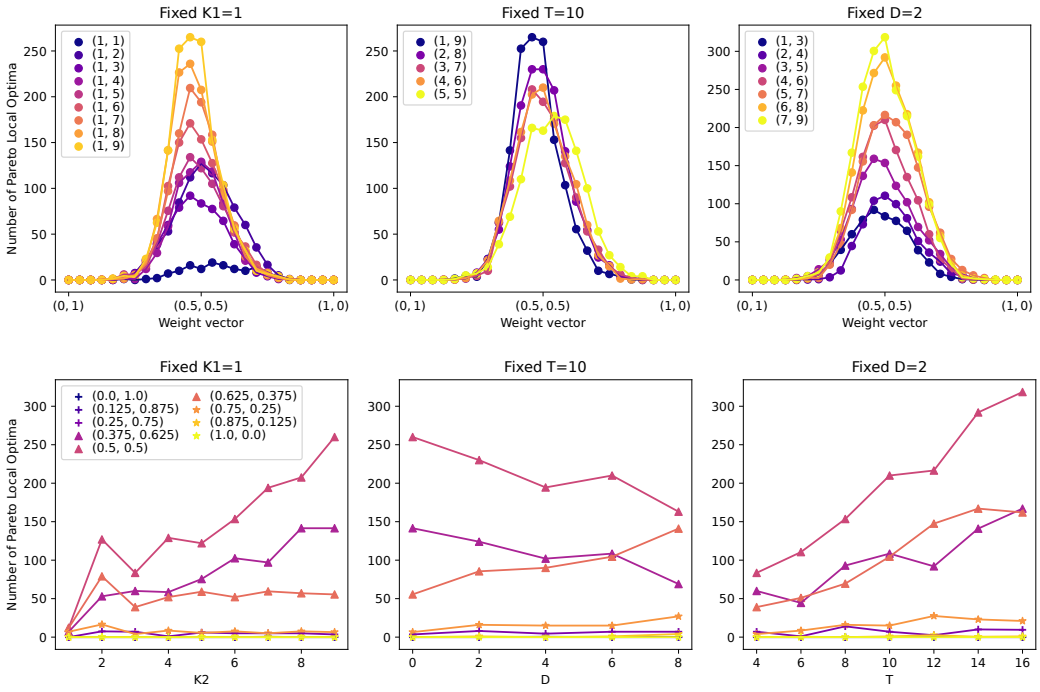


Figure 5.6: Distribution of the Pareto local optima solutions with $\mu = 25$ uniformly distributed weight into the objective space with an angular decomposition (first row). Number of Pareto local optima per weight vectors (second row). Left sub-figures are for fixed $K_1 = 1$. Center sub-figures are for fixed $T = K_1 + K_2 = 10$. Right sub-figures are for fixed $D = K_2 - K_1 = 2$.

observe the peak around the weight vector $(0.5, 0.5)$. Notice that the tail values indicate that Pareto local optima solutions span only a restricted part of

the objective space. More importantly, the number of Pareto local optima solutions increases consistently for higher values of $K_2$, hence indicating that the global difficulty of the multi-objective landscape also increases with $K_2$. In the center part of Figure 5.6, we see the Pareto local optima distribution for instances with a fixed $T = K_1 + K_2$ but a varying difference of the degree of heterogeneity $D = K_2 - K_1$. We clearly see that increasing the heterogeneity degree $D$ results overall in an increase (roughly up to 30%) of Pareto local optima solutions at the most centered vectors. Interestingly, the position of the highest peak deviates lightly from the most centered vector, and the increase of Pareto local optima solutions is not systematic, in the sense that for some weight vectors, the corresponding number of Pareto local optima solutions decreases as shown in the bottom sub-figure. In the last right part of Figure 5.6, we show the Pareto local optima distribution for instances with a fixed degree of heterogeneity $D = K_2 - K_1$, but a varying value of $D = K_1 + K_2$. As expected, the number of Pareto local optima increases with $T$, which seems to be the most impacting parameter. As commented previously, a heterogeneous landscape still contains more Pareto local optima than a homogeneous one with the same $T$, however this increment is not as large as when $T$ increases.

**Distribution based on Scalar Decomposition:** The distributions observed in the previous section inform about where the Pareto local optima solutions lay over different regions of the objective space. However, they do not fully inform about the difficulty of reaching some particular regions of the objective space. For this purpose, we complement the Pareto local optima distribution analysis by adopting a different decomposition technique. The proposed approach is similar to decomposition-based features but focuses on the metric distribution over ordered sub-problems instead of the aggregated multi-objective feature. More specifically, we still consider a set of $\mu$ uniformly distributed weight vectors $w^i$, $i \in \{1, \ldots, \mu\}$. Using the Chebyshev scalarizing function to define $\mu$ single-objective sub-problems, we then consider studying the single-objective local optima for each sub-problem. We recall that since we are considering small size instances in this section, this can be performed by full enumeration of the search space. In particular, the number of local optima is known to provide useful information about the ruggedness of a single-objective landscape. This is shown in Figure 5.8 for different heterogeneous settings and for $\mu = 25$ sub-problems.

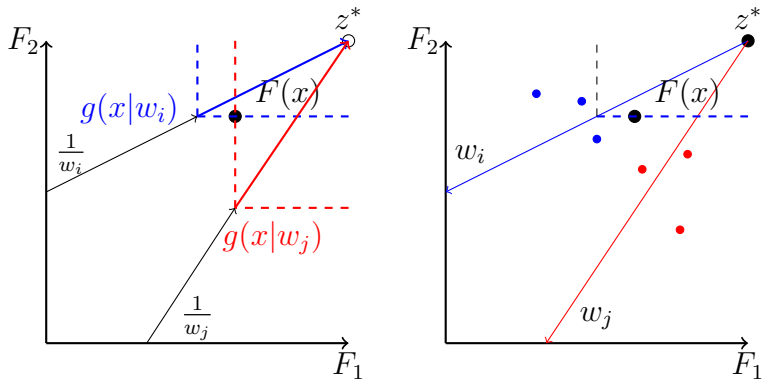


Figure 5.7: Illustration of the distribution using a Chebyshev scalarizing function.

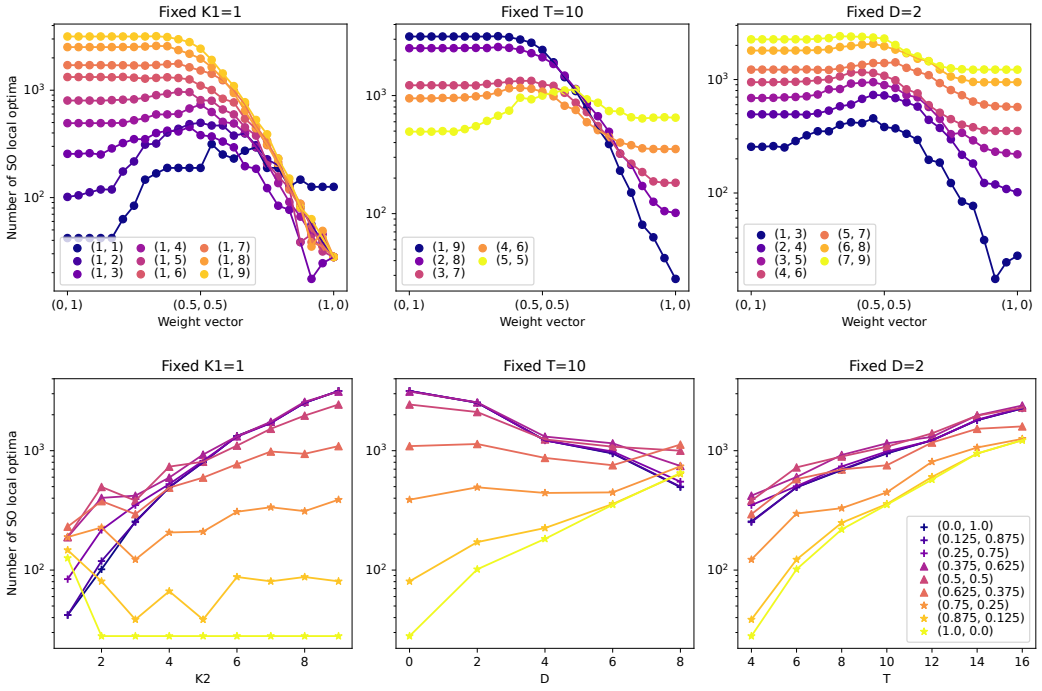


Figure 5.8: Average number of local optima solutions using Chebyshev scalarizing function with $\mu = 25$ uniformly distributed weight. The first row is for varying the weight vector, and the second row is for varying $K_2$. Left sub-figures are for fixed $K_1 = 1$. Center sub-figures are for fixed $T = K_1 + K_2 = 10$. Right sub-figures are for fixed $D = K_2 - K_1 = 2$.

A number of observations can be extracted from Figure 5.8. Firstly, as can be seen in the left sub-figures, the number of LO is not symmetric with respect to the central weight vector, which is to contrast with the previous observations for the Pareto local optima solutions. On the one hand, the higher $K_2$ the higher the number of LO solutions. On the other hand, the closer a sub-problem is to the second objective (with higher $K_2$), the more the number of local optima. This indicates that the difficulty of sub-problems increases gradually from the easiest objective (weight vector $(1, 0)$ with $K_1 = 1$) to the hardest one (weight vector $(0, 1)$ with $K_2 > 1$). Notice however that the number of local optima solutions with respect to sub-problems that are very close to the first objective (with $K_1 = 1$) do not seem to be significantly influenced by the difficulty of the second objective (with $K_2$). Besides, when the total degree of interactions $T$ is fixed (sub-figures in the middle), the number of local optima solutions depends not only on the considered sub-problem, but also on the degree of heterogeneity $D$. The more homogeneous an instance, the smaller the difference in the number of local optima solutions between the first/easiest and last/hardest sub-problem.

Finally, when the degree of heterogeneity $D$ is fixed (sub-figures on the right), the number of local optima solutions as a function of sub-problems have a similar shape independently of $T$, but with a different magnitude. Notice that for these instances with similar fixed $D$, the central sub-problems have a slightly larger number of local optima solutions, even compared to the hardest objective with $K_2$ interactions. Remember that sub-problems are defined with respect to the Chebyshev scalarizing function, which could explain why the combination of a hard and an easy problem, respectively with $K_1$ and $K_2$ interactions, results in a single-objective landscape with an even larger number of local optima solutions, hence being a priori more difficult to search.

In the next section, we propose to use some features in an attempt to better quantify the variations in the distribution of Pareto local optima and local optima solutions in the objective space by means of simple numerical values.

**Aggregating the Distribution of Local Optima into Multi-objective Features:** In the following, we propose to aggregate the previous distributions of Pareto local optima and local optima solutions into single numerical values, forming a set of multi-objective global features that can be computed by full enumeration of the search space. We follow the same approach as for decomposition-based features used in Chapter 2. The resulting features complement the existing ones while aiming at specifically capturing the prop-

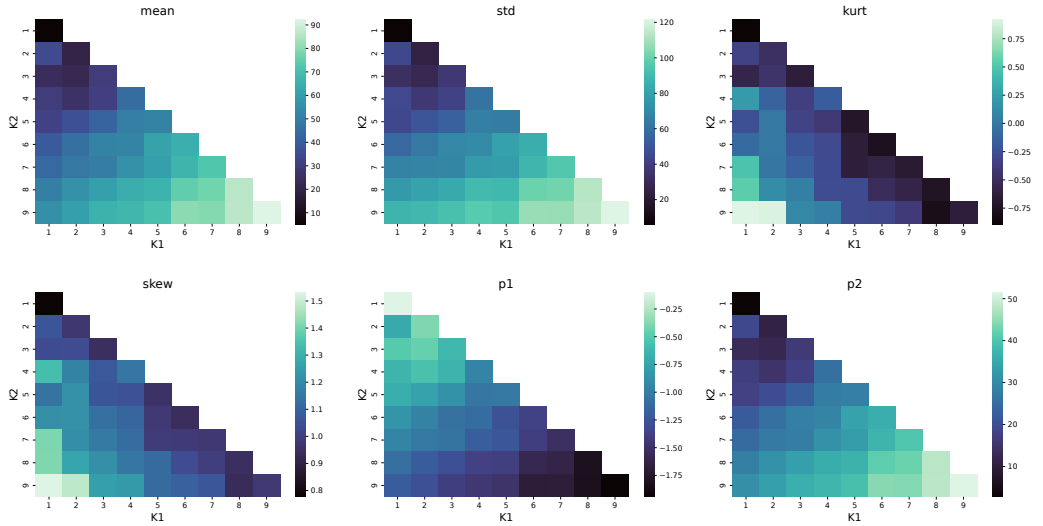


Figure 5.9: Aggregated multi-objective features as a function of $K_1$ and $K_2$ using the distribution of the number of Pareto local optima with respect to an angular decomposition.

erties of heterogeneous landscapes. More precisely, the number of Pareto local optima and local optima solutions computed previously are aggregated along the weight vectors using the following descriptive statistics: the mean, the standard deviation, the kurtosis, the skewness, as well as the first and second coefficient of a second order polynomial regression to fit the single-objective features as a function of the ordered indices of sub-problem weight vectors. We thus end up with six features with respect to Pareto local optima, as shown in Figure 5.9, and six features with respect to local optima, as shown in Figure 5.10.

Several observations can be extracted for the general tendency of the feature values as a function of instance heterogeneity. First, the mean number of Pareto local optima and the mean number of local optima (top left of Figures 5.9 and 5.10) increase with $K_1$ and $K_2$ (and hence with $T$). However, the tendency for the standard deviation (top center of Figures 5.9 and 5.10) is different depending on whether Pareto local optima or local optima are considered. In fact, the standard deviation of the number of Pareto local optima solutions increases with the total degree of interaction $T$. By contrast, the standard deviation of local optima solutions is mostly impacted by the degree of heterogeneity $D$. The kurtosis feature is mainly intended to capture the tails of the underlying distribution. As can be seen in the top right of Figure 5.9, the Pareto local optima distribution has a negative kurtosis with a small de-

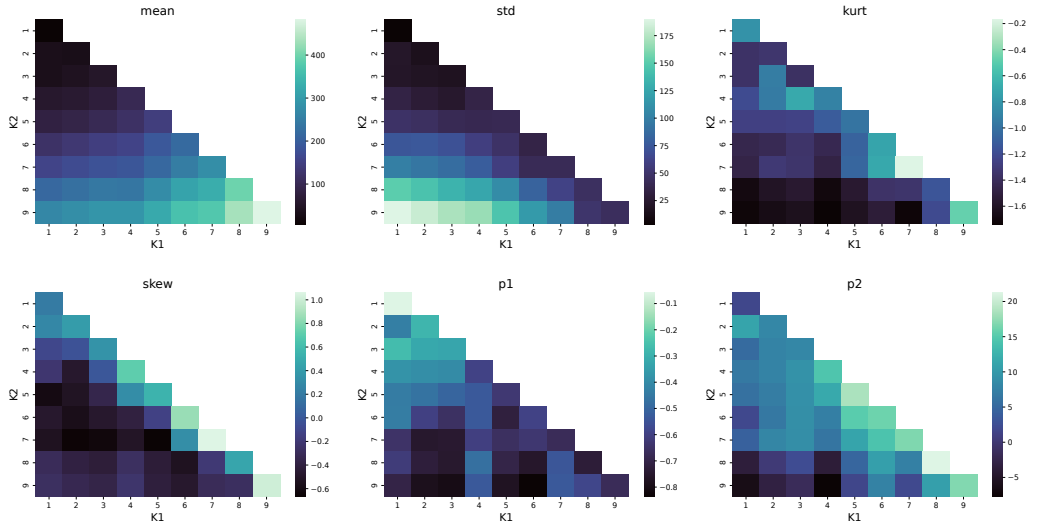


Figure 5.10: Aggregated multi-objective features as a function of $K_1$ and $K_2$ using the distribution of the number of local optima with respect to a Chebyshev decomposition.

gree of heterogeneity $D$, meaning that most Pareto local optima solutions are around the center area of the objective space. By contrast, highly heterogeneous instances have high positive kurtosis values, indicating more Pareto local optima solutions in the boundaries. As for local optima solutions (top right of Figure 5.10), all kurtosis values are found to be negative. It is however still possible to discriminate between homogeneous instances to heterogeneous due to the asymmetric shape of the distribution observed previously. This is actually captured by the skewness statistic, which is a standard measure of the asymmetry of a distribution, showing that for heterogeneous instances, the peak in the number of Pareto local optima peak is not perfectly centered in the objective space, but slightly shifted towards the second (hardest) objective. The last statistics are the first and second coefficients of a second order polynomial regression (bottom center and right of Figures 5.9 and 5.10). They are found to be mostly related to the total number of interactions $T$ when computed with respect to Pareto local optima solutions. However, these two features do not show a smooth behavior when computed with respect to local optima solutions.

**Summary and Outlook:** From the observations provided by the angular- and scalar-based decomposition techniques and the underlying multi-objective

global features, we can conclude that Pareto local optima and local optima solutions are distributed differently depending on the heterogeneity parameters $D$ and $T$ (equivalently, $K_1$ and $K_2$). They also highly depend on the considered region of the objective space. On the one hand, the number of Pareto local optima solutions is higher in the center of the objective space, while being overall positively correlated to $T$. On the other hand, the number of local optima solutions (with respect to the Chebyshev scalarizing function) is asymmetric with respect to the central objective space area, with the regions near the hardest objective containing overall more local optima solutions compared to the central area, except for specific combinations of $D$ and $T$. Interestingly, this suggests that identifying the Pareto set underlying heterogeneous landscapes might imply different levels of difficulty, not only from a global perspective as a function of the heterogeneity parameters $D$ and $T$, but also from a more local perspective, in the sense that some areas of the Pareto front might be more difficult to approach than others. It is worth noticing that the analysis conducted in the previous section did not target the difficulty of searching a given heterogeneous landscape, but rather the intrinsic properties of the landscape obtained by fully enumerating the search space. Hence, although being informative on the impact of objective heterogeneity on the multi-objective landscape, the observations obtained in the scope of our first empirical analysis need to be consolidated with further considerations taking into account large scale problems, as well as existing algorithms. This is precisely the aim of the next section.

## 5.5 Feature-based Analysis for Large Problems

In this section, we consider larger problems for which it is not possible to fully enumerate the search space. Similarly to small problems, our main goal is to provide new insights on how the heterogeneity in objective difficulty can impact the landscape, hopefully confirming our previous observations. For this purpose, we rely on existing state-of-the-art landscape features that can be computed by sampling the search space.

### 5.5.1 Multi-objectives Features Behavior on Heterogeneous MNK-landscapes

In Table 5.2, we recall the subset of landscape features that we consider in this chapter. Among decomposition-based features, we considered two main feature subsets, namely, (i) the number of improving neighbors (in.*.*) nor-

Table 5.2: Summary of the considered multi-objective landscape features.

| Set | name | description | |
|---|---|---|---|
| **Decomposition** | in.d.r | Number of improving neighbors proportional to the dimension | Chapter 2 |
| | fd.d.r | proportional fitness difference with the neighboring solutions | |
| | law.r | Length of adaptive walk | |
| **Indicator** | hv.s | hypervolume covered by solutions | [55] |
| | hvd.s | hypervolume difference with the neighboring solutions | |
| | nhv.s | hypervolume of the neighboring solutions | |
| **Dominance** | inf.s | proportion of neighbors dominated by the current solution | [55] |
| | sup.s | proportion of neighbors dominating the current solution | |
| | inc.s | proportion of neighbors incomparable to the current solution | |

d ∈ {avg,sd}, r ∈ {avg,sd,p1,p2}, s ∈ {avg,sd,r1}

malized by the total number of neighbors (i.e., the size of the neighborhood), and (ii) the fitness difference between a solution and its neighbors (fd.*.*) normalized by the fitness of the actual solution. Additionally, the length of a single-objective adaptive walk (law.*) performed independently for each subproblem is considered (scalar walk). For each decomposition-based feature, we consider the mean (avg) and the standard deviation (sd). Features are aggregated with all originally proposed aggregation functions: the mean (avg), the standard deviation (sd), and the first and second coefficient of a second order polynomial regression (p1,p2).

Among indicator-based features, we consider: the hypervolume of a solution (hv.*), the difference of hypervolume between a solution and its neighbors (hvd.*) and the hypervolume of neighboring solutions (nhv.*). For the dominance-based features, we use: the proportion of neighbors dominated by the current solution (inf.*), the proportion of neighbors dominating the current solution (sup.*) and the proportion of neighbors incomparable to the current solution (inc.*). Functions used for the indicator and dominance-based features are: the mean (*.avg), the standard deviation (*.sd) and the first auto-correlation coefficient (*.r1) [96].

Features were computed on 81 heterogeneous MNK-landscapes of dimension $n = 50$ with $K_1$ and $K_2$ varying in the set $\{1, 2, \ldots, 9\}$ as in Section 5.3. All three feature sets were computed with a random walk of size $\ell = 200$. The

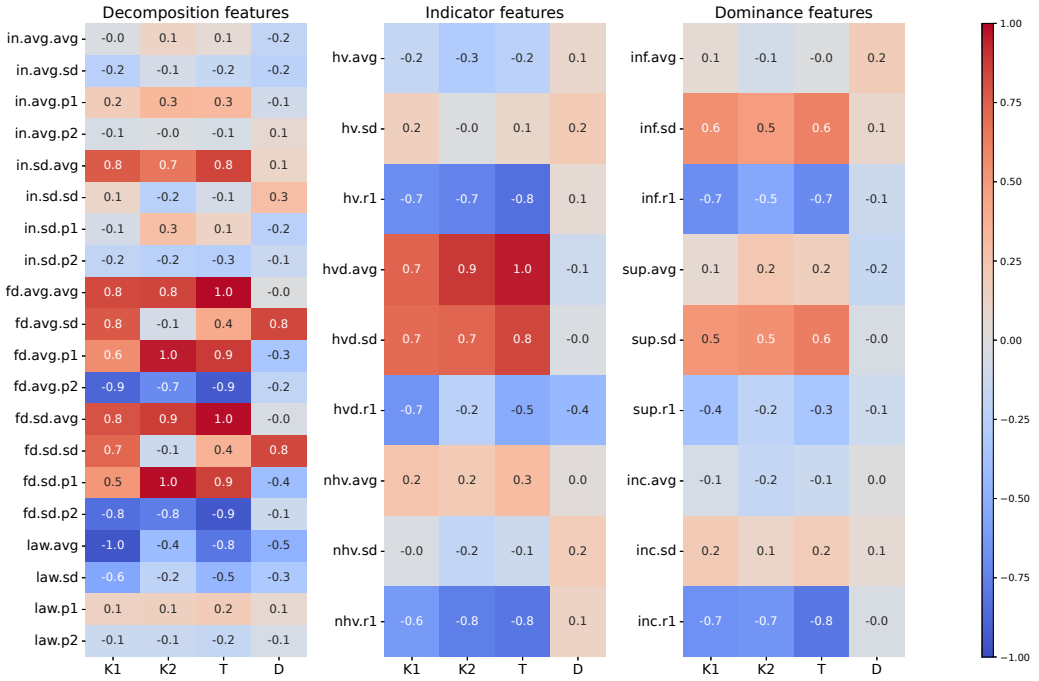


Figure 5.11: Pairwise Spearman correlation between features and benchmark parameters. Decomposition, indicator, and dominance-based features are from left to right respectively.

features of the first set use a Chebyshev decomposition with 25 uniformly distributed weight vectors and the reference point $z^\star$ is computed over the entire sample for each instance such that $z_i^\star > F_i(x)$, $i \in \{1, \ldots, m\}$ for each solution $x$ in the sample. The length of the adaptive walk was computed using a single adaptive walk for each of the 25 weight vectors, and a dynamic reference point which is updated each time the neighborhood of the current solution is evaluated.

**Correlation Analysis:** We start our feature-based analysis by reporting in Figure 5.11 the pairwise Spearman correlation matrices between the considered features and the instance parameters $K_1$, $K_2$, $T$ and $D$. One can see a number of interesting observations. In the case of decomposition-based features, all features from the fitness difference subset (fd.*.*) are strongly correlated with at least one of the instance parameters. On the one hand, most features in this subset are correlated (or anti-correlated in the case of fd.*.p2) with $K_1$ , $K_2$ and $T$. On the other hand, features that are aggregated with a standard

deviation at the second level (fd.avg.sd and fd.sd.sd) are the only ones among the fd.*.* subset that appear to be correlated with the degree of heterogeneity $D$. Features relating to the number of improving neighbors in.*.* are not very correlated with the instance parameters, except for the feature in.sd.avg, which is found to be correlated with $K_1$, $K_2$ and $T$. As for the third subset of decomposition-based features relating to the length of the adaptive walk (law.*), we see that the average length (law.avg) is anti-correlated to $K_1$ and $T$, meaning that increasing the global ruggedness of the landscape $T$ translates into shorter adaptive walks, which is likely because it becomes easier to fall into a local optima. Among the indicator-based features, those using the first auto-correlation coefficient as an aggregation statistic (hv.r1, hvd.r1 and nhv.r1) are all anti-correlated with $K_1$. Actually, the hypervolume (hv.r1) and the neighborhood hypervolume (nhv.r1) features are also anti-correlated with $K_2$ and $T$. The mean hypervolume difference and its standard deviation (hv.avg, hvd.sd) are however strongly correlated with $K_1$, $K_2$ and $T$. Finally, it is interesting to remark that the dominance-based features appear to have a relatively low correlation with the instance parameters compared to the two other sets, except for inf.r1 and inc.r1. Overall, it is interesting to remark that few features seem to be (positively or negatively) correlated to the degree of heterogeneity $D = K_2 - K_1$, which was shown to be a key element in our small size instance analysis. Interestingly, only the decomposition-based fd.avg.sd and fd.sd.sd features are found to be highly correlated to $D$.

**Feature-based Prediction of Heterogeneity:** To provide additional information about the ability of the considered features in capturing instance heterogeneity, we consider the task of predicting the initial instance parameters ($K_1$, $K_2$, $T$, and $D$) using the so-computed feature values. For this purpose, we use a single-output decision tree regressor with default parameters. We actually trained four models using all features sets to predict respectively (i) $K1$, (ii) $K2$, (iii) $T$, and (iv) $D$. For each model, we compute the mean $R^2$ and the standard deviation over 50 independent repetitions of training as we have 81 problem instances, results are computed on test data using a K-fold cross-validation with $K = 9$. This is summarized in Table 5.3. Besides, we report, in Figure 5.12, the fifteen features of the most important normalized rank for each model using the mean Gini impurity as a measure of importance.

We see that all models reach a mean $R^2$ value of at least 0.94, showing that features are relatively successful in characterizing the original parameters of the landscape. We can however report some differences between the accuracy between the degree of heterogeneity $D$ and the others parameters as it is harder

Table 5.3: Mean $R^2$ and standard deviation of multi-output random forest regression models tasked to predict all benchmark parameters using as input the union of decomposition, indicator and dominance-based features sets.

| Parameter | $K1$ | $K2$ | $T$ | $D$ |
|---|---|---|---|---|
| Mean $R^2$ | 0.96 | 0.98 | 0.98 | 0.94 |
| Standard deviation $R^2$ | 0.007 | 0.004 | 0.003 | 0.008 |

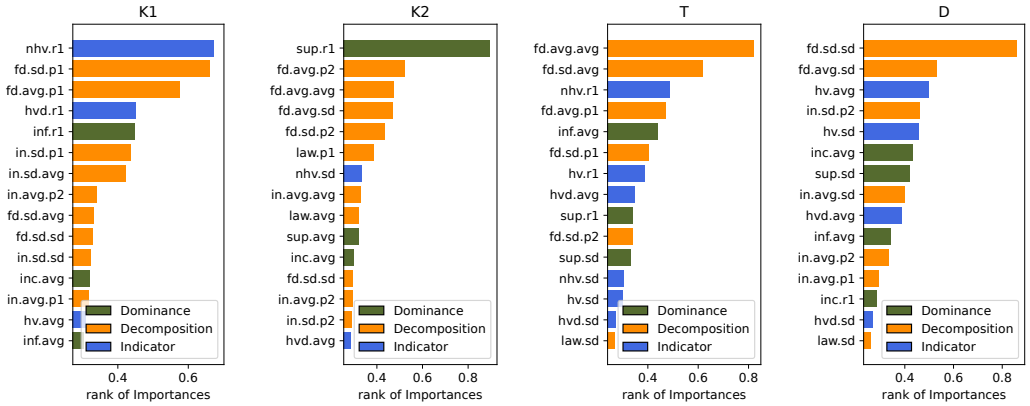


Figure 5.12: Mean feature importance of random forest regression learning models tasked to predict benchmark parameters using as input the union of decomposition, indicator and dominance-based features sets.

to predict by a small margin. Both original parameters of the benchmark, $K1$ and $K2$ are accurately predicted by decision trees, but features inherently behind those predictions are not the same. From Figure 5.12, we see that the most important features for $K1$ are nhv.r1 and fd.sd.p1 which have a similar rank. For $K2$ the most important feature is sup.r1 from the dominance set. The main factor between features differences in $K1$ and $K2$ decision trees is the asymmetry of used heterogeneous MNK-landscapes instances as we considered $K1 < K2$. Alternative parameters $T$ and $D$ reach respectively a mean $R^2$ of 0.98 and 0.94 with a standard deviation around 0.003 for $T$ and 0.008 for $D$. Features lead to fairly accurate predictions for both parameters. Notice that the most important features (fd.avg.avg, fd.sd.avg) for $T$ and (fd.sd.sd, fd.avg.sd) for $D$ are both the same decomposition features variation with a different aggregation function, and they were found to be correlated respectively with $T$ and $D$. Interestingly, all feature sets are of importance to predict each parameter.

### 5.5.2 Impact of Heterogeneity on Algorithm Performance

In this section, we conclude our analysis by studying the impact of heterogeneity on multi-objective algorithm behavior. For this purpose, we consider two well-established multi-objective evolutionary algorithms as detailed next.

**Experimental Setup:** We consider MOEA/D [62] and NSGA-II [26] introduced in Chapter 1, which are respectively based on decomposition and dominance. The two algorithms are implemented following the standard default setting as described in their respective original papers. For both algorithms, we use a population of size 100, and a uniform crossover followed by a uniform bit-flip mutation where each bit is flipped with a rate of $1/n$. For MOEA/D, we use the Chebyshev scalarizing function with 100 uniform weight vectors, a sub-problem neighborhood size of 6, a probability of mating with a solution in the neighborhood $\delta = 0.9$ and the maximal number of solutions replaced by each offspring solution $nr = 2$ [104, 62]. The maximal budget is set to $10^6$ evaluations per run. Each algorithm is independently executed 20 times on each instances of dimension $n = 50$ with $K_1$ and $K_2$ varying in the set $\{1, 2, \ldots, 9\}$.

**Algorithm Performance:** To assess algorithm performance, we consider two conventional quality indicators from multi-objective optimization [109]. The hypervolume relative deviation, as shown in the first row of Figure 5.13, is the relative deviation between the hypervolume [107] of the approximation found by an algorithm and the hypervolume of the reference set (obtained by merging all runs of all algorithms for a specific instance). The second indicator is the $R2$ indicator [37], as shown in the second row of Figure 5.13. The $R2$ assess the relative quality between a front and the ideal front. Given a set of $\mu$ weight vectors $W$, a set of evaluated solutions $A$ and the ideal front $PF$, the indicator is defined as:

$$R2(A, W, PF) = \frac{1}{\mu} \sum_{w_i \in W} \left( \max_{r \in PF}(g(r, w_i)) - \max_{x \in A}(g(x, w_i)) \right) \tag{5.5}$$

The smaller the value of the $R2$ indicator, the better the front approximation. It is interesting to notice that the $R2$ indicator has similarities with the distribution statistics/features described in Sections 5.4, as it is based on a Chebyshev scalarizing function with 100 weight vectors.

Although Figure 5.13 shows a clear difference in the relative performance of MOEA/D and NSGA-II, which we will comment later, we first comment

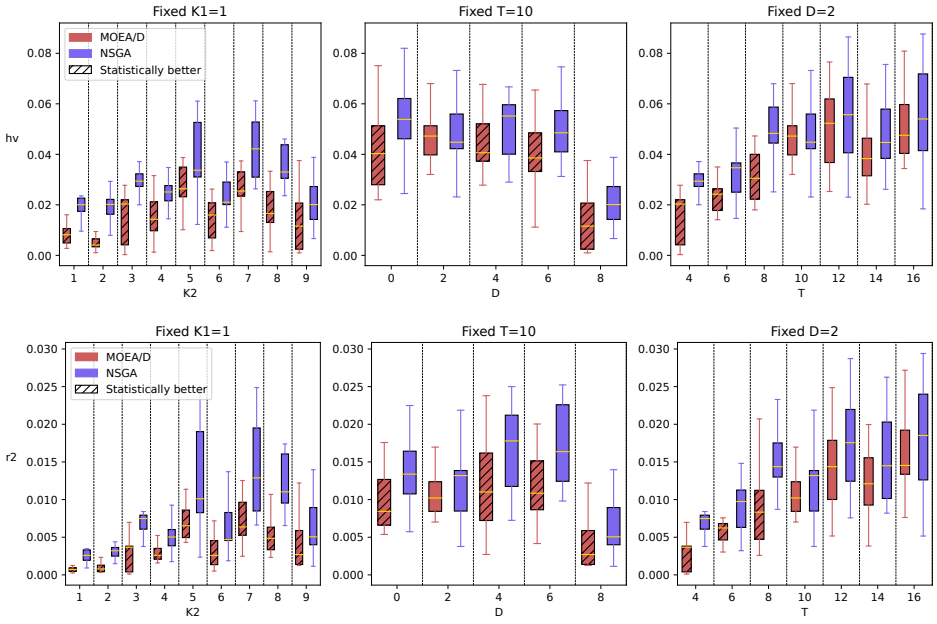


Figure 5.13: Hypervolume relative deviation (first row) and R2 indicators of the approximations obtained by MOEA/D and NSGA-II on heterogeneous MNK-landscapes.

that both algorithms interestingly have a similar behavior as a function of the considered instances. For the instances with $K_1 = 1$ (left sub-figures), the hypervolume and the $R2$ indicators increase with $K_2$ before slightly decreasing for $K_2 > 7$. Although the average indicator-values decrease when $K_2$ is high, the standard deviation of indicators appear to be relatively high. For the instance with fixed $T = 10$ (sub-figures in the middle), the indicator-values are relatively high, which indicates that both algorithms are struggling to find a good approximation set, except when $K_1 = 1$ and $K_2 = 9$. Interestingly, the homogeneous instance with $K_1 = K_2 = 5$ seems to be as difficult to solve than heterogeneous instances.

We observe that the impact of the number of variable interactions on the difficulty of the objective function is not to be thought as linear. In fact, the relative difficulty of solving an NK-landscape with $k = 2$ compared to an instance with $k = 1$, is believed to be considerably greater than the relative difficulty of solving an instance with $k = 3$ compared to an instance with $k = 2$. This may explain the observed difference in performance between a completely heterogeneous instance ($K_1 = 1, K_2 = 9$) and the other instances with $T = 10$. When the heterogeneity is fixed at $D = 2$, the approximation

sets obtained by both algorithms appear to be far from the reference set, as indicated by the relatively high indicator-values. The greater the value of $T$, the less likely the algorithms are to find a good approximation. Interestingly, the indicator-values (and subsequently the solving difficulty) seem to increase moderately with $T$ when $D$ is fixed.

As mentioned above, although both algorithms exhibit similar behavior on heterogeneous MNK-landscapes, MOEA/D significantly outperforms NSGA-II on most instances. Using a Wilcoxon statistical test with a $p$-value of 0.05 with Bonferroni correction, the approximations obtained by MOEA/D are statistically better than those of NSGA-II, except for instances with $T \geq 10$ and $D = 2$. The easier the instance, the statistically better MOEA/D over NSGA-II. When the overall difficulty is too high (high $T$ values), both algorithms struggle to find a good approximation set.

**Chebyshev Deviation to the Pareto front:** To better understand the algorithms' behavior according to the heterogeneity of the objectives, we conduct a fine-grained analysis of the distance between approximate solutions found by an algorithm with respect to the reference set. We recall that, as the Pareto front cannot be computed for large problems, we consider the reference set as the aggregation of non-dominated solutions from all runs of both algorithms. Then, we consider a *normalized* Chebyshev scalarizing function over 100 sub-problems with respect to 100 uniformly distributed weight vectors. More precisely, to avoid any bias in the original Chebyshev scalar values, the normalized Chebyshev function uses, instead of the original fitness value of a solution $x$, the following normalization:

$$F_{i,\ normalized}(x) = \frac{z_i^* - F_i(x)}{z_i^* - z_i^{nad}} \tag{5.6}$$

where $F_i$ is objective $i$ and $z^{nad} = (z_1^{nad}, \ldots, z_M^{nad})$ is a nadir point, i.e., $z_i^{nad} \leq F_i(x)\ \forall x \in \mathcal{X},\ i \in \{1, \ldots, m\}$.

For each algorithm run and each (single-objective) sub-problem corresponding to the weight vector $w_i$, we define the normalized Chebyshev difference $\mathcal{D}(w_i|Q)$ between the approximation $Q$ found by the algorithm and the reference set $R$, as follows:

$$\mathcal{D}(w_i|Q) = \frac{\min_{x \in Q}(g(x|w_i)) - \min_{x \in R}(g(x|w_i))}{\min_{x \in R}(g(x|w_i))} \tag{5.7}$$

Notice that the higher $\mathcal{D}(w_i|Q)$, the further the solution found by the algorithm for sub-problem $w_i$ from the best-found solution. In fact, averaging

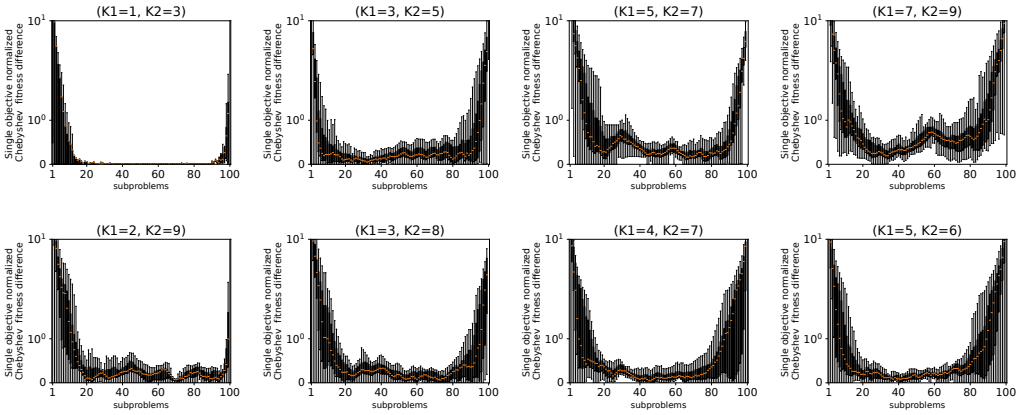


Figure 5.14: Single-objective normalized Chebyshev deviation between MOEA/D and the approximated ideal front for instances with $D = 2$ in the first row and $T = 11$ in the second row

equation (5.7) for all sub-problems $w_i$, $i \in \{1, \ldots, \mu\}$ exactly matches the $R3$ indicator from [37].

The values of the so-obtained Chebyshev differences are reported in Figure 5.5.2 and 5.5.2, respectively, for MOEA/D and NSGA-II. For the clarity of the presentation, we do not show all possible configurations of $K_1$ and $K_2$, but only a representative subset of instances with a fixed total degree of interaction $T = 11$, and a fixed degree of heterogeneity $D = 2$. Additionally, all figures are presented in the Appendix 6.2. As can be clearly observed, the general tendency of the Chebyshev differences is similar for both algorithms as a function of instance characteristics. When the total degree of interaction is relatively low ($K_1 = 1, K_2 = 3$), we observe two peaks corresponding to the extreme weight vectors. The left peak is slightly larger than the right one as the second objective gets harder. However, both algorithms manage to reach the reference set (or to be very close) in the regions corresponding to weight vectors between 20 and 80. When the total degree of interaction $T$ increases, the Chebyshev difference increases for weight vectors corresponding to the center of the objective space, and we observe wider peaks on the extremities as it gets harder to approach the Pareto front. The heterogeneity $D$ also affects the Chebyshev difference for extreme weights vectors. For heterogeneous instances with $K_1 \leq 3$, there is a tighter peak towards the easiest objective, and the more homogeneous the instance, the smaller the difference between extreme weight vectors.

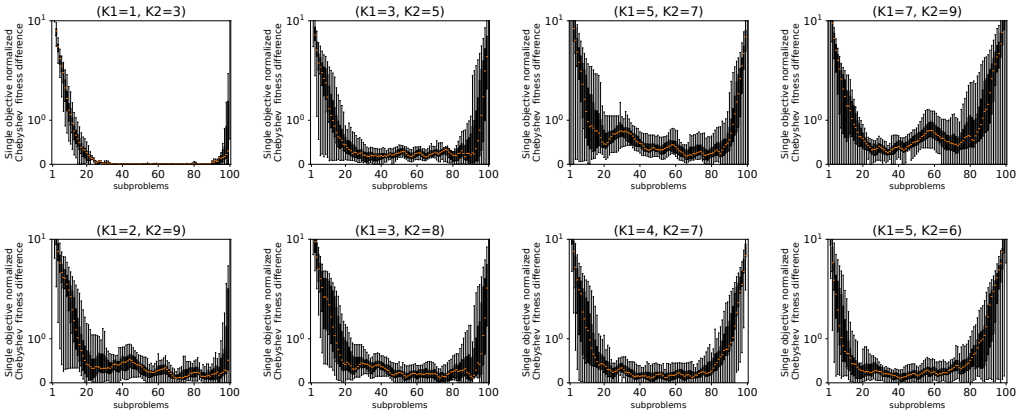


Figure 5.15: Single-objective normalized Chebyshev deviation between NSGA-II and the approximated ideal front for instances with $D = 2$ in the first row and $T = 11$ in the second row

**Features-based Performance Prediction:** We conclude our analysis by studying how accurate the previously considered landscape features are for the purpose of algorithm performance prediction. For this purpose, we proceed as in Section 5.5.1, and we train different regression trees to predict the average hypervolume relative deviation obtained for each algorithm using all feature sets as input. We report the average $R^2$ and the standard deviation of 50 independent repetitions of training with K-fold cross-validation with $K = 9$ as we have 81 problem instances. Additionally, we consider using parameters ($K1$, $K2$, $T$, $D$) as input for decision trees to compare with features accuracy. The achieved mean $R^2$ values are reported in Table 5.4, and the most important features are shown in Figure 5.16.

All regression trees reach a relatively high $R^2$ around 0.9. This means that 90% of the variance in hypervolume values are explained by the model, and thus by the considered landscape features. Performances prediction models trained with benchmark parameters reach an $R^2$ of 0.92 for predicting MOEA/D hypervolume deviation and 0.91 for NSGA-II which is slightly higher than when using features as input. The features' importance presents few distinctions between both algorithms. The two most important features for MOEA/D are fd.avg.p1 and fd.sd.p1. Both were found correlated with $K1$ and $K2$ and are respectively the second and third most important features to predict NSGA-II performances. In the latter, the most important features is nhv.r1 from the indicator subset. Lastly, each features subset have at least one features among the fifteen most important. This shows that each subset

Table 5.4: Mean $R^2$ and standard deviation of multi-output random forest regression models for predicting algorithm performances independently using as input benchmark parameters and the union of decomposition, indicator and dominance-based features sets.

| | Parameters | | Features | |
|---|---|---|---|---|
| Algorithm | MOEA/D | NSGA-II | MOEA/D | NSGA-II |
| Mean $R^2$ | 0.92 | 0.91 | 0.91 | 0.89 |
| Standard deviation $R^2$ | 0.007 | 0.01 | 0.012 | 0.011 |

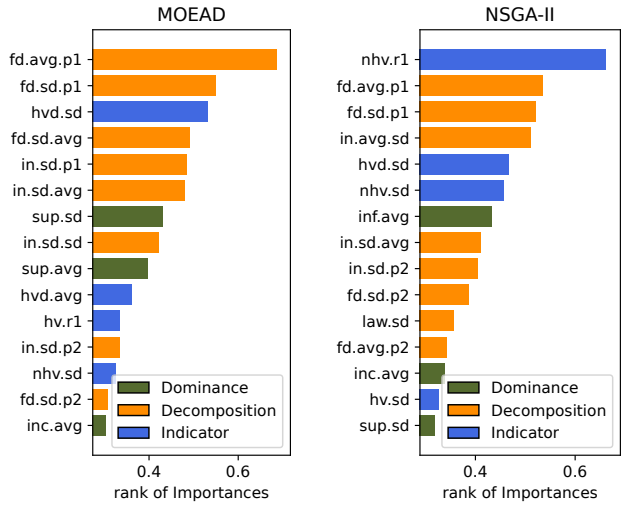


Figure 5.16: Mean feature importance of random forest regression models for predicting algorithm performances independently using as input the union of decomposition, indicator and dominance-based features sets.

is of importance. The most important features appear to be from both the indicator set and the decomposition set. Dominance-based features seem to play a less important role when it comes to predict algorithm performance for heterogeneous MNK-landscapes. This is to contrast with homogeneous MNK-landscapes, where dominance-based features were found to be important for performance prediction [55].

## 5.6 Conclusion

In this chapter, we investigated the properties of multi-objective combinatorial optimization problems, exposing different degrees of heterogeneity in terms of objective difficulty. For small problems, we analyzed the impact of heterogeneity on the multi-objective landscape using novel decomposition techniques informing about the distribution of local optima in the objective space. In particular, we found that heterogeneity introduces local differences in the landscape, increasing the number of Pareto local optimal solutions and single-objective optima mapping to specific regions of the objective space. For large problems, we studied the ability of the three sets of multi-objective landscape features to capture heterogeneity, namely decomposition-based features proposed in Chapter 2 and indicator-based and dominance-based features introduced in Chapter 3. In particular, we showed that decomposition-based and notably fitness difference features are relatively well-suited compared to dominance- and indicator-based feature. Finally, we investigated the impact of heterogeneity on two well-established multi-objective evolutionary algorithms, i.e., MOEA/D and NSGA-II. We found that, although one algorithm outperforms the other, both algorithms behave similarly depending on the degree of heterogeneity. In particular, increasing the difficulty of an objective without changing the other makes it harder for an algorithm to approach the Pareto front in the region close to the hardest objective's optimum.

Our findings also suggest a number of challenging research questions. In fact, heterogeneity in terms of objective difficulty was studied to a very small extent, and our work is to be viewed as a first step toward more extensive studies on the subject. In particular, through the different landscape analysis provided in this chapter, it appears that heterogeneity has a local impact on the shape of the Pareto front and on the difficulty of reaching specific regions. In this respect, it is relatively challenging to design global features that can capture this aspect, and any progress in this direction would be a nice contribution. Besides, we have focused on the bi-objective case, for which standard decomposition-based statistics were relatively well-suited to capture heterogeneity. Considering scenarios with more objectives is challenging from at least two perspectives. First, the heterogeneity between (pairwise) objectives can be thought in different ways, hence possibly leading to a broad range of possible heterogeneous problem types. This raises the challenging question of whether there are some heterogeneous scenarios that are more difficult to solve than others. Second, it is unclear whether the considered landscape features are still accurate in capturing the degree of heterogeneity and problem difficulty as accurately as in the bi-objective case. In this respect, more in-

vestigations are required to design new features dedicated to characterizing heterogeneity when dealing with an increasing number of objectives. Finally, our algorithm performance analysis suggests that some regions of the objective space are consistently more difficult to reach than others. As such, algorithms able to take into account the objective heterogeneity can lead to significantly better performance. In particular, adaptive techniques that can accommodate the search to the landscape heterogeneity are worth investigating in the future, especially in a practical black-box scenario, where the degree of heterogeneity might not be known beforehand.

# Chapter 6

# General Conclusion

Solving multi-objective combinatorial optimization problems is a challenging task. As finding the exact Pareto set is in general very difficult to achieve, especially in black-box optimization, a large body of the literature is devoted to the design of efficient algorithms computing an approximation of the Pareto set. In particular, multi-objective evolutionary algorithms constitute an effective option adopted in a number of settings. However, the performance of any optimization algorithm depends heavily on the characteristics of the problem instance being solved, and evolutionary algorithms stand for no exception. In this context, fitness landscape analysis provides tools and techniques to capture the characteristics of a problem instance, and hence, to gain a better fundamental understanding of what makes an instance difficult to solve by a given algorithm. In other words, fitness landscape analysis is of special interest in order to map algorithm performance with problem characteristics.

In this thesis, we were mainly interested in developing new fundamental tools for multi-objective fitness landscape analysis, and their use for solving high-level optimization tasks. As such, we have described a number of contributions organized in four main chapters: (1) New techniques based on decomposition in order to define novel multi-objective landscape features, (2) A landscape-aware approach for determining the effectiveness of an algorithm according to a given multi-objective problem instance, (3) A detailed analysis of the cost and stability of the designed landscape features, (4) A new multi-objective optimization benchmark focusing on the heterogeneity among the objectives.

In the reminder of this chapter, we first summarize in more detail the main contributions of this thesis. Then, we discuss some open research questions.

## 6.1 Contributions

**Multi-objective landscape analysis based on decomposition:** In our first set of contributions (Chapter 2), we have described a new methodology to compute multi-objective features. The proposed approach is based on decomposing the original multi-objective problem into several single-objective sub-problems. Each sub-problem implies a different landscape, so that a number of single-objective landscape features can be computed accordingly. Such single-objective (sub-problem) landscape features are finally aggregated to form novel multi-objective features. The so-obtained decomposition-based multi-objective features have been investigated empirically on an extensive set of bi-objective pseudo-Boolean problems based on a well-established multi-objective combinatorial optimization benchmark. Our empirical findings show that the proposed decomposition-based features can accurately and successfully capture useful information about the considered multi-objective landscapes.

**Solving high-level optimization tasks:** In our second set of contributions (Chapter 3), we have considered the solving of two high-level challenging tasks. The first task consists in predicting the benchmark global parameters, which aims at analyzing the effectiveness of features in characterizing a given problem instance. The second task consists in addressing the algorithm selection problem, given a portfolio of competing algorithms and a given problem instance to solve. To tackle both tasks, we follow a standard supervised machine learning approach where a learning model is trained with some landscape features given as input. We considered the proposed decomposition-based features, as well as the main existing multi-objective features taken from the specialized literature, namely indicator-based and dominance-based features. Our empirical findings showed that a feature-based automated approach is extremely accurate for both tasks, and that the designed decomposition-based features are highly valuable.

**Analysis of feature computation cost:** In the third set of contributions (Chapter 4), we pushed our analysis further on the properties of existing multi-objective landscape features and their ability to automatically select the best algorithm. As such, we focused on the impact of the budget used to compute and extract landscape features. We studied two simple cost-adjustable walks as sampling methods, by controlling two parameters in the sampling phase, namely, the length of walks and the proportion of the explored neighborhood. As expected, the overall efficiency improves with the size of the sample used for

feature computation. However, we find significant differences between the feature values depending on the sampling parameters. Some features are mostly affected by the length of the walk while being insensitive to the proportion of explored neighbors, whereas the situation is reversed for other features. We thereby used the feature value convergence rate and the correlation with the benchmark parameters in order to highlight the complex interaction between feature accuracy and sampling cost. Besides, specifically with respect to automated algorithm selection, we proposed an indicator to measure performance that take into account the cost of sampling. Our experimental findings show that, by accurately setting the sampling parameters, it is possible to find efficient trade-offs in terms of sampling cost and features quality when performing the prediction and algorithm selection task. In other words, with a well parameterized sampling method, using the prediction model remains a far better option, even when counting with the cost of sampling, compared to a simple feature-free strategy that consists in always picking the algorithm that performs best in average.

**Heterogeneous multi-objective NK-landscapes:** The last set of contributions (Chapter 5) consists in studying the heterogeneity in multi-objective optimization. We introduced heterogeneous MNK-landscapes as a new benchmark exposing different levels of difficulty between the objectives. To understand how such heterogeneity impacts the landscape properties, we considered both a full enumeration approach for small problem instances, and a sampling-based approach to compute features for large problem instances. For small instances, our experimental findings indicate that the number and the distribution of Pareto local optimal solutions is strongly influenced by objective heterogeneity. For large instances, using a set of decomposition-based features, we were able to highlight the difference between landscapes with different degrees of heterogeneity. While the proposed features succeeded in grasping the overall structure of the considered landscapes, we showed that there is still a need to design new features to fully capture the objective heterogeneity. Finally, for further understanding the impact of heterogeneity in terms of solving difficulty, we investigated the performance of two state-of-the-art algorithms. Overall, we showed that although the two algorithms have a different performance, they consistently share the same behavior as a function of the heterogeneity degree, in the sense that, it was harder for both of them to approximate efficiently the Pareto front on the boundary near the most difficult objective.

## 6.2 Perspectives

**Analyzing other multi-objective optimization problems:** In our work, we restricted our experiments to bi-objective landscapes. This is mainly to keep our investigations more focused, and to be able to perform an extensive and comprehensive analysis. A straightforward extension of our work would be to consider optimization problems with more than two objectives. Notice that although most of the proposed landscape features are still well-defined regardless to the number of objectives, it is still worth to investigate settings with a variable number of objectives. This is to further confirm the relative accuracy of the different considered features, especially when adopting an automated feature-based approach for the algorithm selection problem. More generally, other discrete representations, such as permutations, should be considered in the future. Besides, the general principle guiding the design of the proposed decomposition-based features can also apply for continuous domains, in the sense that any existing single-objective feature could in theory be used as a basis for defining an aggregated multi-objective feature. However, since the sampling strategies used for extracting features with respect to continuous optimization problems could be seemingly different from the combinatorial case, a lot of efforts remain to be accomplished before confirming the applicability and the accuracy of our decomposition-based approach for multi-objective continuous optimization.

**A systematic approach to feature design and integration:** Designing landscape features is a difficult task. On the one hand, different approaches might be adopted depending on the target optimization context. For instance, one may want to design features with respect to a particular problem, a particular domain, a particular portfolio of algorithms, a particular class of solving methods, etc. Hence, it is not always clear what constitutes a "good set" of landscape features, and what formal properties it should verify independently of the optimization context or task under consideration. On the other hand, there are different design options that can typically lead to a relatively large number of possible features. In the case of our work, we were specifically interested in black-box multi-objective combinatorial optimization problems, while considering pseudo-boolean domains and a range of standard state-of-the-art algorithms. This actually represents a relatively restricted setting, which has the advantage of keeping our investigations focused. Nevertheless, even in such a setting, designing informative cheap features was not easy to achieve. Besides, the considered set of possible multi-objective features is still relatively large, which is mainly because different multi-objective paradigms could be

adopted, as well as different samplings, parameters, and feature options. As such, we tried to provide a systematic understanding of what makes a feature meaningful and accurate for a given task. In this respect, using feature importance as provided, e.g., by the underlying random forest models was in particular very useful. A challenging future research direction is to set up a more systematic methodology, to help the design and integration of new features, which is as independent as possible of the optimization setting or task. The general purpose would be to derive a more principled development loop, where one is able to design a new feature, to study its properties, and to consider integrating it in an existing set for accomplishing some specific purpose. Such a loop could even be thought to operate in an automated manner, for example by considering some feature subset selection techniques, or some feature classification techniques, depending on the target optimization setting. We believe that such a research path has to be accomplished in a cooperative manner, for instance by providing the community with advanced generic software tools, and easy (online) interfaces for accessing, analyzing and integrating, landscape features and their corresponding data.

**Improving automated algorithm selection:** In our work, we have shown that a feature-based approach for addressing the automated algorithm selection problem is particularly accurate. However, a number of important questions remain open. A straightforward extension of our work is to consider a larger set of multi-objective algorithms. One interesting question is to study more specifically what kind of features are more accurate for a given algorithm, or a particular class of algorithms. In fact, besides allowing us to gain a more in-depth understanding of what makes an algorithm successful, this will eventually help designing new dedicated features, guided by the specific behavior or trajectory of the considered algorithms. Another question is to consider settings where the maximum affordable budget is not known beforehand. In such a setting, new techniques have to be derived in order to take the ability of a given feature into account to capture the anytime behavior of an algorithm. This is actually a very challenging question that requires new learning models and new landscape-guided automated selection methodologies. Finally, the automated selection methodology depends heavily on the accuracy of the problem instance training set. An important question, which is related to more general benchmarking aspects, is to be able to derive, in a more systematic manner, a diverse set of training instances that cover accurately the feature space. Different methods exist for this purpose, and it is worth investigating their relevancy at the aim of designing more powerful

automated algorithm selection approaches.

**Addressing heterogeneity in multi-objective optimization:** The last chapter of this thesis was devoted to study heterogeneity in multi-objective optimization. This is actually one important aspect that was addressed to a relatively small extent in the past. A number of important questions remain open, and different research perspectives can be considered in this context. In the following, we discuss two important aspects.

Firstly, the proposed heterogeneous MNK-landscapes can be extended following different perspectives. For example, it remains unclear what is the impact of using more than two heterogeneous objectives on the underlying multi-objective landscape, neither how such a setting implies different degrees of solving difficulties. One might also consider different ways of controlling heterogeneity when defining the single-objective problems. For instance, one could manage to have seemingly different interactions between the variables depending on the objective and analyze how this could impact heterogeneity. More generally, when moving to different models, representations or domains, one may wonder what could be a representative benchmark to model heterogeneity, and whether the same features designed in our work are still meaningful when considering different benchmark settings.

Secondly, to the best of our knowledge, there are no state-of-the-art algorithm that were specifically designed to deal with objectives having a different degree of ruggedness. Our experimentation using NSGA-II and MOEA/D showed that both algorithms consistently face the same challenge of approximating the front in the boundary of the most difficult objective. This suggests that enhanced methods could be designed by typically adapting the search with respect to the regions of the objective space that are more difficult to reach. This would in fact constitute a nice achievement in the relatively large panel of multi-objective evolutionary algorithms. Besides, the information gained from the designed landscape features is expected to provide new research paths towards solving techniques accurately designed for heterogeneous multi-objective optimization problems.

# Appendix

## Summary of all multi-objective landscape features

During the thesis, a total of 125 features were considered and designed. The features vary according to their type, the observed metric, the computed statistics and the aggregation functions in the case of the decomposition-based features. Each of the following tables correspond to a type of features : dominance-based, indicator-based, decomposition-based and the length of adaptive walk, which can be or not considered in any of the previous sets.

Table 6.1: Table of dominance-based features.

| Dominance-based features [55] | | |
|---|---|---|
| Observed value | statistical function computed from the sample | feature name |
| Proportion of dominated neighbors | mean | inf.avg |
| | standard deviation | inf.sd |
| | first auto-correlation coefficient | inf.r1 |
| | kurtosis | inf.kr |
| | skewness | inf.sk |
| Proportion of dominating neighbors | mean | sup.avg |
| | standard deviation | sup.sd |
| | first auto-correlation coefficient | sup.r1 |
| | kurtosis | sup.kr |
| | skewness | sup.sk |
| Proportion of incomparable neighbors | mean | inc.avg |
| | standard deviation | inc.sd |
| | first auto-correlation coefficient | inc.r1 |
| | kurtosis | inc.kr |
| | skewness | inc.sk |
| Proportion of locally non-dominated neighbors | mean | lnd.avg |
| | standard deviation | lnd.sd |
| | first auto-correlation coefficient | lnd.r1 |
| | kurtosis | lnd.kr |
| | skewness | lnd.sk |
| Proportion of supported locally non-dominated neighbors | mean | lsupp.avg |
| | standard deviation | lsupp.sd |
| | first auto-correlation coefficient | lsupp.r1 |
| | kurtosis | lsupp.kr |
| | skewness | lsupp.sk |

Table 6.2: Table of indicator-based features.

| Indicator-based features [55] | | |
|---|---|---|
| Observed value | statistical function computed from the sample | feature name |
| Solution's hypervolume | mean | hv.avg |
| | standard deviation | hv.sd |
| | first auto-correlation coefficient | hv.r1 |
| | kurtosis | hv.kr |
| | skewness | hv.sk |
| Hypervolume of the neighborhood | mean | nhv.avg |
| | standard deviation | nhv.sd |
| | first auto-correlation coefficient | nhv.r1 |
| | kurtosis | nhv.kr |
| | skewness | nhv.sk |
| Difference of hypervolume between a solution and its neighbors | mean | hvd.avg |
| | standard deviation | hvd.sd |
| | first auto-correlation coefficient | hvd.r1 |
| | kurtosis | hvd.kr |
| | skewness | hvd.sk |

Table 6.3: Table of additional features which do not belong to other subsets.

| Observed value | statistical function computed from the sample | feature name |
|---|---|---|
| Length of adaptive walks (when computed from a multi-objective adaptive walk) | mean | law.avg |
| | standard deviation | law.sd |
| | first auto-correlation coefficient | law.r1 |
| | kurtosis | law.kr |
| | skewness | law.sk |

Table 6.4: Table of the Maximum distance between improving ordered sub-problems, subset of decomposition-based features .

| Maximum distance between improving ordered sub-problem [22] | |
|---|---|
| statistical function computed from the sample | feature name |
| Mean | spd.avg |
| standard deviation | spd.sd |
| Minimum | spd.min |
| Maximum | spd.max |
| First auto-correlation [46] | spd.r1 |
| Kurtosis [110] | spd.kr |
| Skewness [110] | spd.sk |
| metric normalization : proportional to the number of sub-problems $\mu$ | |

Table 6.5: Table of the fitness values subset of features (decomposition-based features set).

| Fitness value features [21, 22] | | |
|---|---|---|
| statistical function computed from the sample | aggregation function between sub-problems | feature name |
| Mean | mean | fv.avg.avg |
| | standard deviation | fv.avg.sd |
| | $1^{st}$ polynomial coefficient | fv.avg.p1 |
| | $2^{nd}$ polynomial coefficient | fv.avg.p2 |
| Standard deviation | mean | fv.sd.avg |
| | standard deviation | fv.sd.sd |
| | $1^{st}$ polynomial coefficient | fv.sd.p1 |
| | $2^{nd}$ polynomial coefficient | fv.sd.p2 |
| Minimum | mean | fv.min.avg |
| | standard deviation | fv.min.sd |
| | $1^{st}$ polynomial coefficient | fv.min.p1 |
| | $2^{nd}$ polynomial coefficient | fv.min.p2 |
| Maximum | mean | fv.max.avg |
| | standard deviation | fv.max.sd |
| | $1^{st}$ polynomial coefficient | fv.max.p1 |
| | $2^{nd}$ polynomial coefficient | fv.max.p2 |
| First auto-correlation [46] | mean | fv.r1.avg |
| | standard deviation | fv.r1.sd |
| | $1^{st}$ polynomial coefficient | fv.r1.p1 |
| | $2^{nd}$ polynomial coefficient | fv.r1.p2 |
| Kurtosis [110] | mean | fv.kr.avg |
| | standard deviation | fv.kr.sd |
| | $1^{st}$ polynomial coefficient | fv.kr.p1 |
| | $2^{nd}$ polynomial coefficient | fv.kr.p2 |
| Skewness [110] | mean | fv.sk.avg |
| | standard deviation | fv.sk.sd |
| | $1^{st}$ polynomial coefficient | fv.sk.p1 |
| | $2^{nd}$ polynomial coefficient | fv.sk.p2 |
| metric normalization : all original multi-objective values are normalized with $\hat{f}_i(x) = \frac{f_i(x) - \min_{\forall x \in \mathcal{X}} f_i(x)}{\max_{\forall x \in \mathcal{X}} f_i(x) - \min_{\forall x \in \mathcal{X}} f_i(x)}$ | | |

Table 6.6: Table of the fitness distance subset of features (decomposition-based features set).

| Fitness distance features [21, 22] | | |
|---|---|---|
| statistical function computed from the sample | aggregation function between sub-problems | feature name |
| Mean | mean | fd.avg.avg |
| | standard deviation | fd.avg.sd |
| | $1^{st}$ polynomial coefficient | fd.avg.p1 |
| | $2^{nd}$ polynomial coefficient | fd.avg.p2 |
| Standard deviation | mean | fd.sd.avg |
| | standard deviation | fd.sd.sd |
| | $1^{st}$ polynomial coefficient | fd.sd.p1 |
| | $2^{nd}$ polynomial coefficient | fd.sd.p2 |
| Minimum | mean | fd.min.avg |
| | standard deviation | fd.min.sd |
| | $1^{st}$ polynomial coefficient | fd.min.p1 |
| | $2^{nd}$ polynomial coefficient | fd.min.p2 |
| Maximum | mean | fd.max.avg |
| | standard deviation | fd.max.sd |
| | $1^{st}$ polynomial coefficient | fd.max.p1 |
| | $2^{nd}$ polynomial coefficient | fd.max.p2 |
| First auto-correlation [46] | mean | fd.r1.avg |
| | standard deviation | fd.r1.sd |
| | $1^{st}$ polynomial coefficient | fd.r1.p1 |
| | $2^{nd}$ polynomial coefficient | fd.r1.p2 |
| Kurtosis [110] | mean | fd.kr.avg |
| | standard deviation | fd.kr.sd |
| | $1^{st}$ polynomial coefficient | fd.kr.p1 |
| | $2^{nd}$ polynomial coefficient | fd.kr.p2 |
| Skewness [110] | mean | fd.sk.avg |
| | standard deviation | fd.sk.sd |
| | $1^{st}$ polynomial coefficient | fd.sk.p1 |
| | $2^{nd}$ polynomial coefficient | fd.sk.p2 |
| metric normalization : proportional fitness difference $fd(x, x') = \frac{\lvert F(x) - F(x') \rvert}{F(x)}$ | | |

Table 6.7: Table of the Improving neighbors subset of features (decomposition-based features set).

| Improving neighbors features [21, 22] | | |
|---|---|---|
| statistical function computed from the sample | aggregation function between sub-problems | feature name |
| Mean | mean | fd.avg.avg |
| | standard deviation | in.avg.sd |
| | $1^{st}$ polynomial coefficient | in.avg.p1 |
| | $2^{nd}$ polynomial coefficient | in.avg.p2 |
| Standard deviation | mean | in.sd.avg |
| | standard deviation | in.sd.sd |
| | $1^{st}$ polynomial coefficient | in.sd.p1 |
| | $2^{nd}$ polynomial coefficient | in.sd.p2 |
| Minimum | mean | in.min.avg |
| | standard deviation | in.min.sd |
| | $1^{st}$ polynomial coefficient | in.min.p1 |
| | $2^{nd}$ polynomial coefficient | in.min.p2 |
| Maximum | mean | in.max.avg |
| | standard deviation | in.max.sd |
| | $1^{st}$ polynomial coefficient | in.max.p1 |
| | $2^{nd}$ polynomial coefficient | in.max.p2 |
| First auto-correlation [46] | mean | in.r1.avg |
| | standard deviation | in.r1.sd |
| | $1^{st}$ polynomial coefficient | in.r1.p1 |
| | $2^{nd}$ polynomial coefficient | in.r1.p2 |
| Kurtosis [110] | mean | in.kr.avg |
| | standard deviation | in.kr.sd |
| | $1^{st}$ polynomial coefficient | in.kr.p1 |
| | $2^{nd}$ polynomial coefficient | in.kr.p2 |
| Skewness [110] | mean | in.sk.avg |
| | standard deviation | in.sk.sd |
| | $1^{st}$ polynomial coefficient | in.sk.p1 |
| | $2^{nd}$ polynomial coefficient | in.sk.p2 |
| metric normalization : proportional to the number of dimension $n$ | | |

Table 6.8: Table of the length of Adaptive walk subset of features (decomposition-based features set with Scalar sampling method).

| length of Adaptive walk [21, 22] | | |
|---|---|---|
| statistical function computed from the sample (if more than one walk is used) | aggregation function between sub-problems | feature name |
| Mean | mean | law.avg.avg |
| | standard deviation | law.avg.sd |
| | $1^{st}$ polynomial coefficient | law.avg.p1 |
| | $2^{nd}$ polynomial coefficient | law.avg.p2 |
| Standard deviation | mean | law.sd.avg |
| | standard deviation | law.sd.sd |
| | $1^{st}$ polynomial coefficient | law.sd.p1 |
| | $2^{nd}$ polynomial coefficient | law.sd.p2 |
| Minimum | mean | law.min.avg |
| | standard deviation | law.min.sd |
| | $1^{st}$ polynomial coefficient | law.min.p1 |
| | $2^{nd}$ polynomial coefficient | law.min.p2 |
| Maximum | mean | law.max.avg |
| | standard deviation | law.max.sd |
| | $1^{st}$ polynomial coefficient | law.max.p1 |
| | $2^{nd}$ polynomial coefficient | law.max.p2 |
| First auto-correlation [46] | mean | law.r1.avg |
| | standard deviation | law.r1.sd |
| | $1^{st}$ polynomial coefficient | law.r1.p1 |
| | $2^{nd}$ polynomial coefficient | law.r1.p2 |
| Kurtosis [110] | mean | law.kr.avg |
| | standard deviation | law.kr.sd |
| | $1^{st}$ polynomial coefficient | law.kr.p1 |
| | $2^{nd}$ polynomial coefficient | law.kr.p2 |
| Skewness [110] | mean | law.sk.avg |
| | standard deviation | law.sk.sd |
| | $1^{st}$ polynomial coefficient | law.sk.p1 |
| | $2^{nd}$ polynomial coefficient | law.sk.p2 |

## Additional statistics over Heterogeneous MNK-landscapes


Figure 6.1: Additional Statistics computed over heterogeneous MNK landscapes of dimension 16. Number of Pareto fronts per instance, computed using the non-dominated sort algorithm 3 (top left). Average Number of local optima per instance (top right). Average Number of connected components (bottom left). Proportional deviation of single-objective local optima between each objectives (bottom right).

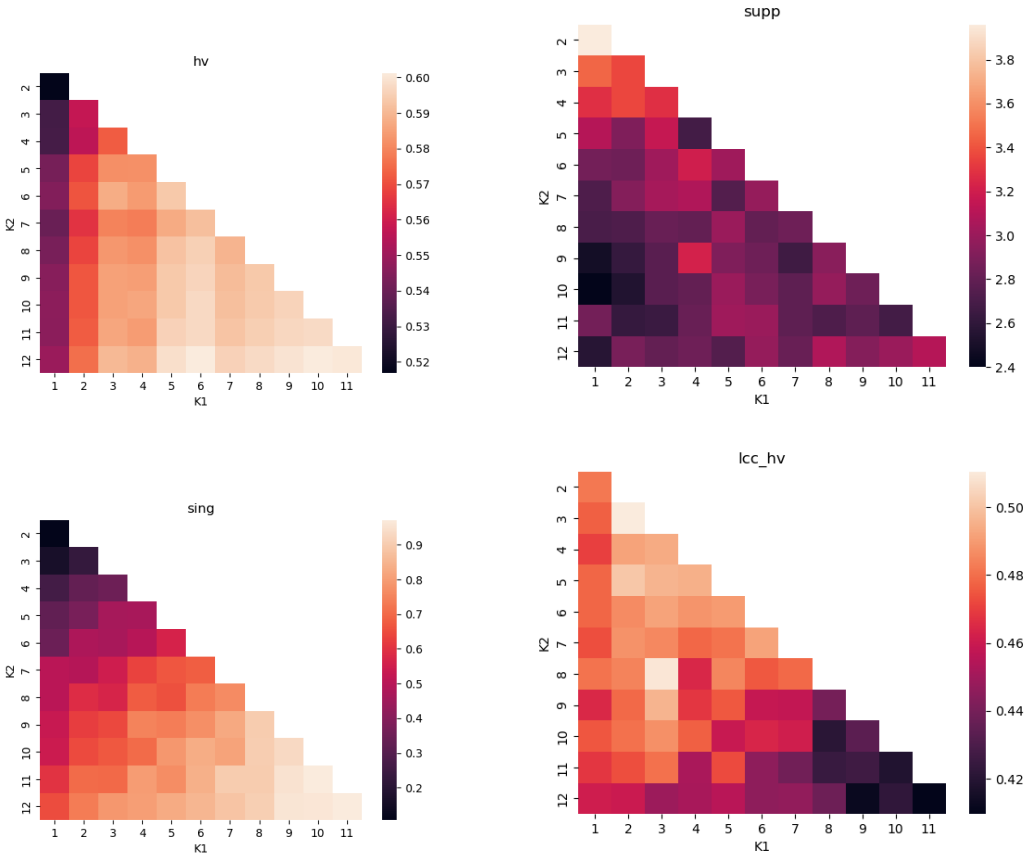


Figure 6.2: Additional Statistics computed over heterogeneous MNK landscapes of dimension 16. Hypervolume of the Pareto front (bottom left). Number of supported solutions (top right). Proportion of isolated Pareto-optimal solutions (bottom left). Hypervolume of the largest connected component (bottom right).

## Chebyshev Deviation to the Pareto Front for Heterogeneous MNK-landscapes

Figure 6.3: Single-objective normalized Chebyshev deviation between MOEA/D and the approximated ideal front for all instances, sorted by increasing Degree of heterogeneity ($D = 0$ at the top) and by total degree of interaction (lower $T$ on the left).

Figure 6.4: Single-objective normalized Chebyshev deviation between NSGA-II and the approximated ideal front for all instances, sorted by increasing Degree of heterogeneity ($D = 0$ at the top) and by total degree of interaction (lower $T$ on the left).

# Bibliography


[1] Aguirre, H., and Tanaka, K. Insights and properties of multiobjective MNK-landscapes. In *Proceedings of the 2004 Congress on Evolutionary Computation CEC-2004* (Portland, Oregon, 2004), IEEE Press, pp. 196–203.

[2] Aguirre, H., and Tanaka, K. Insights on properties of multiobjective MNK-landscapes. *Forensic Science International-genetics - FORENSIC SCI INT-GENET 1* (07 2004), 196 – 203 Vol.1.

[3] Aguirre, H. E., and Tanaka, K. Working principles, behavior, and performance of MOEAs on MNK-landscapes. *European Journal of Operational Research 181*, 3 (2007), 1670–1690.

[4] Aldous, D., and Fill, J. A. Reversible markov chains and random walks on graphs, 2002.

[5] Allmendinger, R., Handl, J., and Knowles, J. Multiobjective optimization: When objectives exhibit non-uniform latencies. *European Journal of Operational Research 243*, 2 (2015), 497–513.

[6] Allmendinger, R., and Knowles, J. 'hang on a minute': Investigations on the effects of delayed objective functions in multiobjective optimization. In *International Conference on Evolutionary Multi-Criterion Optimization* (2013), Springer, pp. 6–20.

[7] Allmendinger, R., and Knowles, J. Heterogeneous objectives: state-of-the-art and future research. *CoRR abs/2103.15546* (2021).

[8] Auger, A., Bader, J., Brockhoff, D., and Zitzler, E. Theory of the hypervolume indicator: optimal $\mu$-distributions and the choice of the reference point. In *Foundations of Genetic Algorithms (FOGA 2009)* (Orlando, Florida, United States, Jan. 2009).

[9] Back, T., Fogel, D. B., and Michalewicz, Z. *Handbook of Evolutionary Computation*, 1st ed. IOP Publishing Ltd., GBR, 1997.

[10] Bartz-Beielstein, T., Doerr, C., Bossek, J., Chandrasekaran, S., Eftimov, T., Fischbach, A., Kerschke, P., López-Ibáñez, M., Malan, K. M., Moore, J. H., Naujoks, B., Orzechowski, P., Volz, V., Wagner, M., and Weise, T. Benchmarking in optimization: Best practice and open issues. *CoRR abs/2007.03488* (2020).

[11] Belkhir, N. *Per Instance Algorithm Configuration for Continuous Black Box Optimization*. Theses, Université Paris Saclay (COmUE), Nov. 2017.

[12] Beume, N., Naujoks, B., and Emmerich, M. Sms-emoa: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research 181*, 3 (2007), 1653–1669.

[13] Bianchi, L., Dorigo, M., Gambardella, L. M., and Gutjahr, W. J. A survey on metaheuristics for stochastic combinatorial optimization. *Natural Computing: An International Journal 8*, 2 (jun 2009), 239–287.

[14] Blank, J., and Deb, K. Constrained bi-objective surrogate-assisted optimization of problems with heterogeneous evaluation times: Expensive objectives and inexpensive constraints. In *Proceedings of the Evolutionary Multi-Criterion Optimization (EMO-2021)* (2021), pp. 257–269.

[15] Breiman, L. Random forests. *Machine Learning 45*, 1 (2001), 5–32.

[16] Brockhoff, D., Tusar, T., Auger, A., and Hansen, N. Using well-understood single-objective functions in multiobjective black-box optimization test suites, 2016.

[17] Brockhoff, D., Wagner, T., and Trautmann, H. On the properties of the r2 indicator. In *Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation* (New York, NY, USA, 2012), GECCO '12, Association for Computing Machinery, p. 465–472.

[18] Cheng, R., Jin, Y., Olhofer, M., and Sendhoff, B. A reference vector guided evolutionary algorithm for many-objective optimization. *IEEE Transactions on Evolutionary Computation 20*, 5 (2016), 773–791.

[19] Chugh, T., Allmendinger, R., Ojalehto, V., and Miettinen, K. Surrogate-assisted evolutionary biobjective optimization for objectives with non-uniform latencies. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2018)* (2018), pp. 609–616.

[20] Collautti, M., Malitsky, Y., Mehta, D., and O'Sullivan, B. Snnap: Solver-based nearest neighbor for algorithm portfolios. In *Machine Learning and Knowledge Discovery in Databases* (Berlin, Heidelberg, 2013), H. Blockeel, K. Kersting, S. Nijssen, and F. Železný, Eds., Springer Berlin Heidelberg, pp. 435–450.

[21] Cosson, R., Derbel, B., Liefooghe, A., Aguirre, H., Tanaka, K., and Zhang, Q. Decomposition-based multi-objective landscape features and automated algorithm selection. In *European Conference on Evolutionary Computation in Combinatorial Optimization (Part of EvoStar)* (2021), Springer, pp. 34–50.

[22] Cosson, R., Derbel, B., Liefooghe, A., Vérel, S., Aguirre, H. E., Zhang, Q., and Tanaka, K. Cost-vs-accuracy of sampling in multi-objective combinatorial exploratory landscape analysis. In *GECCO 2022 - Genetic and Evolutionary Computation Conference* (Boston, MA, United States, July 2022), Association for Computing Machinery, pp. 493–501.

[23] Cosson, R., Santana, R., Derbel, B., and Liefooghe, A. Multi-Objective NK Landscapes with Heterogeneous Objectives. In *Proceedings of the Genetic and Evolutionary Computation Conference* (New York, NY, USA, 2022), GECCO '22, Association for Computing Machinery, p. 502–510.

[24] Daolio, F., Liefooghe, A., Verel, S., Aguirre, H., and Tanaka, K. Problem features versus algorithm performance on rugged multiobjective combinatorial fitness landscapes. *Evolutionary Computation 25*, 4 (2017), 555–585.

[25] De Jong, K. A. *An Analysis of the Behavior of a Class of Genetic Adaptive Systems.* PhD thesis, University of Michigan, USA, 1975. AAI7609381.

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

[27] Doerr, B., Le, H. P., Makhmara, R., and Nguyen, T. D. Fast genetic algorithms. *CoRR abs/1703.03334* (2017).

[28] Ehrgott, M. Springer Berlin, Heidelberg, January 2005.

[29] Eiben, A. E., and Smith, J. E. *Evolutionary Computing: The Origins*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2015, pp. 13–24.

[30] Fang, Y., and Li, J. A review of tournament selection in genetic programming. In *Proceedings of the 5th International Conference on Advances in Computation and Intelligence* (Berlin, Heidelberg, 2010), ISICA'10, Springer-Verlag, p. 181–192.

[31] Fonseca, C., Fleming, P., Zitzler, E., Thiele, L., and Deb, K. *Evolutionary Multi-Criterion Optimization: Second International Conference, EMO 2003, Faro, Portugal, April 8–11, 2003. Proceedings*. Springer Berlin, Heidelberg, 04 2003.

[32] Fonseca, C. M., and Fleming, P. J. An overview of evolutionary algorithms in multiobjective optimization. *Evol. Comput. 3*, 1 (mar 1995), 1–16.

[33] Friedrich, T., Kötzing, T., Neumann, F., and Radhakrishnan, A. Theoretical study of optimizing rugged landscapes with the cga. In *Parallel Problem Solving from Nature – PPSN XVII* (Cham, 2022), G. Rudolph, A. V. Kononova, H. Aguirre, P. Kerschke, G. Ochoa, and T. Tušar, Eds., Springer International Publishing, pp. 586–599.

[34] Graff, M., Poli, R., and Moraglio, A. Linear selection. In *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2007, 25-28 September 2007, Singapore* (2007), IEEE, pp. 2598–2605.

[35] Greco, S., Klamroth, K., Knowles, J. D., and Rudolph, G. Understanding complexity in multiobjective optimization (Dagstuhl Seminar 15031). In *Dagstuhl Reports* (2015), vol. 5, Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

[36] H. Frauenfelder, A.R. Bishop, A. Garcia, A. Perelson, P. Schuster, D. Sherrington, and P.J. Swart. Landscape paradigms in physics and biology : Concepts, structures, and dynamics. *Physica D* (1997), 107.

[37] Hansen, M., and Jaszkiewicz, A. *Evaluating the quality of approximations to the non-dominated set.* Institute of Mathematical Modeling, Technical University of Denmark, 1998.

[38] Hansen, N., Finck, S., Ros, R., and Auger, A. Real-Parameter Black-Box Optimization Benchmarking 2009: Noiseless Functions Definitions. Research Report RR-6829, INRIA, 2009.

[39] Hénaux, V., Goëffon, A., and Saubion, F. Evolving fitness landscapes with complementary fitness functions. In *Artificial Evolution* (Cham, 2020), L. Idoumghar, P. Legrand, A. Liefooghe, E. Lutton, N. Monmarché, and M. Schoenauer, Eds., Springer International Publishing, pp. 110–120.

[40] Hotelling, H., and Pabst, M. R. Rank Correlation and Tests of Significance Involving No Assumption of Normality. *The Annals of Mathematical Statistics 7*, 1 (1936), 29 – 43.

[41] Huang, Y., Li, W., Tian, F., and Meng, X. A fitness landscape ruggedness multiobjective differential evolution algorithm with a reinforcement learning strategy. *Applied Soft Computing 96* (2020), 106693.

[42] Igel, C., and Toussaint, M. On classes of functions for which no free lunch results hold. *CoRR cs.NE/0108011* (2001).

[43] Iman, R. L. *Latin Hypercube Sampling.* John Wiley & Sons, Ltd, 2006.

[44] Jacobson, S., and Yücesan, E. Analyzing the performance of generalized hill climbing algorithms. *J. Heuristics 10* (07 2004), 387–405.

[45] Jebari, K., Mediafi, M., and Elmoujahid, A. Parent selection operators for genetic algorithms. *International journal of engineering research and technology 2* (2013).

[46] Kauffman, S. *Origins of Order.* Oxford University Press, 1993.

[47] Kauffman, S., and Levin, S. Towards a general theory of adaptive walks on rugged landscapes. *Journal of Theoretical Biology 128*, 1 (1987), 11–45.

[48] Kerschke, P., Hoos, H. H., Neumann, F., and Trautmann, H. Automated algorithm selection: Survey and perspectives. *Evol. Comput. 27*, 1 (2019), 3–45.

[49] Kerschke, P., Preuss, M., Wessing, S., and Trautmann, H. Low-budget exploratory landscape analysis on multiple peaks models. In *The on Genetic and Evolutionary Computation Conference GECCO* (2016), ACM, pp. 229–236.

[50] Kerschke, P., and Trautmann, H. The r-package flacco for exploratory landscape analysis with applications to multi-objective optimization problems. In *2016 IEEE Congress on Evolutionary Computation (CEC)* (2016), pp. 5262–5269.

[51] Kerschke, P., and Trautmann, H. Automated algorithm selection on continuous black-box problems by combining exploratory landscape analysis and machine learning. *Evol. Comput. 27*, 1 (2019), 99–127.

[52] Knowles, J., and Corne, D. Instance generators and test suites for the multiobjective quadratic assignment problem. In *Evolutionary Multi-criterion Optimization* (2003), Springer, pp. 295–310.

[53] Knowles, J., and Corne, D. Quantifying the effects of objective space dimension in evolutionary multiobjective optimization. In *Evolutionary Multi-Criterion Optimization* (Berlin, Heidelberg, 2007), S. Obayashi, K. Deb, C. Poloni, T. Hiroyasu, and T. Murata, Eds., Springer Berlin Heidelberg, pp. 757–771.

[54] Lamont, G. B., and van Veldhuizen, D. A. Multiobjective evolutionary algorithms: classifications, analyses, and new innovations.

[55] Liefooghe, A., Daolio, F., Verel, S., Derbel, B., Aguirre, H., and Tanaka, K. Landscape-aware performance prediction for evolutionary multiobjective optimization. *IEEE Transactions on Evolutionary Computation 24*, 6 (2020), 1063–1077.

[56] Liefooghe, A., Derbel, B., Verel, S., López-Ibáñez, M., Aguirre, H., and Tanaka, K. On Pareto local optimal solutions networks. In *International Conference on Parallel Problem Solving from Nature* (2018), Springer, pp. 232–244.

[57] Liefooghe, A., Vérel, S., Derbel, B., Aguirre, H. E., and Tanaka, K. Dominance, indicator and decomposition based search for multi-objective QAP: landscape analysis and automated algorithm selection. In *Parallel Problem Solving from Nature - PPSN XVI - 16th International Conference, PPSN 2020, Leiden, The Netherlands, September*

*5-9, 2020, Proceedings, Part I* (2020), T. Bäck, M. Preuss, A. H. Deutz, H. Wang, C. Doerr, M. T. M. Emmerich, and H. Trautmann, Eds., vol. 12269 of *Lecture Notes in Computer Science*, Springer, pp. 33–47.

[58] Liefooghe, A., Vérel, S., Lacroix, B., Zavoianu, A., and McCall, J. A. W. Landscape features and automated algorithm selection for multi-objective interpolated continuous optimisation problems. In *Genetic and Evolutionary Computation (GECCO 2021)* (Lille, France, 2021), ACM, pp. 421–429.

[59] Lindauer, M., van Rijn, J. N., and Kotthoff, L. The algorithm selection competition series 2015-17. *CoRR abs/1805.01214* (2018).

[60] Lipowski, A., and Lipowska, D. Roulette-wheel selection via stochastic acceptance. *Physica A: Statistical Mechanics and its Applications 391*, 6 (2012), 2193–2196.

[61] Malan, K. M. A survey of advances in landscape analysis for optimisation. *Algorithms 14*, 2 (2021), 40.

[62] Marquet, G., Derbel, B., Liefooghe, A., and Talbi, E.-G. Shake them all! Rethinking Selection and Replacement in MOEA/D. *PPSN XIII* (2014), 641–651.

[63] Marquet, G., Derbel, B., Liefooghe, A., and Talbi, E.-G. Shake them all! Rethinking Selection and Replacement in MOEA/D. In *Parallel Problem Solving from Nature – PPSN XIII* (Cham, 2014), T. Bartz-Beielstein, J. Branke, B. Filipič, and J. Smith, Eds., Springer International Publishing, pp. 641–651.

[64] Mersmann, O., Bischl, B., Trautmann, H., Preuss, M., Weihs, C., and Rudolph, G. Exploratory landscape analysis. In *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation* (New York, NY, USA, 2011), GECCO '11, Association for Computing Machinery, p. 829–836.

[65] Morgan, R., and Gallagher, M. Sampling techniques and distance metrics in high dimensional continuous landscape analysis: Limitations and improvements. *IEEE Transactions on Evolutionary Computation 18*, 3 (2014), 456–461.

[66] Nourmohammadzadeh, A., Sarhani, M., and Voss, S. Fitness landscape ruggedness impact on pso in dealing with three variants of

the travelling salesman problem. In *Learning and Intelligent Optimization* (Cham, 2022), D. E. Simos, V. A. Rasskazova, F. Archetti, I. S. Kotsireas, and P. M. Pardalos, Eds., Springer International Publishing, pp. 429–444.

[67] Paquete, L., Schiavinotto, T., and Stützle, T. On local optima in multiobjective combinatorial optimization problems. *Annals of Operations Research 156*, 1 (2007), 83–97.

[68] Paquete, L., and Stützle, T. Clusters of non-dominated solutions in multiobjective combinatorial optimization: An experimental analysis. In *Multiobjective Programming and Goal Programming*. Springer, 2009, pp. 69–77.

[69] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. Scikit-learn: Machine learning in python. *Journal of machine learning research 12*, Oct (2011), 2825–2830.

[70] Pencheva, T., Atanassov, K., and Shannon, A. Modelling of a stochastic universal sampling selection operator in genetic algorithms using generalized nets. 1–7.

[71] Perelson, A., and Kauffman, S. Molecular evolution on rugged landscapes : Protein, rna, and the immune system.

[72] Pisinger, D., and Toth, P. *Knapsack Problems*. Springer US, Boston, MA, 1998, pp. 299–428.

[73] Pruvost, G., Derbel, B., Liefooghe, A., Li, K., and Zhang, Q. On the combined impact of population size and sub-problem selection in MOEA/D. *CoRR abs/2004.06961* (2020).

[74] Radcliffe, N. J., and Surry, P. D. *Fundamental limitations on search algorithms: Evolutionary computing in perspective*. Springer Berlin Heidelberg, Berlin, Heidelberg, 1995, pp. 275–291.

[75] Renau, Q., Doerr, C., Dréo, J., and Doerr, B. Exploratory landscape analysis is strongly sensitive to the sampling strategy. In *Parallel Problem Solving from Nature - PPSN XVI - 16th International Conference, PPSN* (2020), vol. 12270 of *Lecture Notes in Computer Science*, Springer, pp. 139–153.

[76] Renau, Q., Dréo, J., Doerr, C., and Doerr, B. Expressiveness and robustness of landscape features. In *The Genetic and Evolutionary Computation Conference Companion, GECCO* (2019), ACM, pp. 2048–2051.

[77] Rice, J. R. The algorithm selection problem. vol. 15 of *Advances in Computers*. Elsevier, 1976, pp. 65–118.

[78] Richter, H., and Engelbrecht, A., Eds. *Recent Advances in the Theory and Application of Fitness Landscapes*. Emergence, Complexity and Computation. Springer, 2014.

[79] Rizzini, M., Fawcett, C., Vallati, M., Gerevini, A. E., and Hoos, H. H. Portfolio methods for optimal planning: An empirical analysis. In *2015 IEEE 27th International Conference on Tools with Artificial Intelligence (ICTAI)* (2015), pp. 494–501.

[80] Rosa, G. J. M. The elements of statistical learning: Data mining, inference, and prediction by hastie, t., tibshirani, r., and friedman, j. *Biometrics 66*, 4 (2010), 1315–1315.

[81] Saleem, S., Gallagher, M., and Wood, I. Direct Feature Evaluation in Black-Box Optimization Using Problem Transformations. *Evolutionary Computation 27*, 1 (03 2019), 75–98.

[82] Saxena, A., and Pandey, J. Review of crossover techniques for genetic algorithms. 2394–9333.

[83] Schumacher, C., Vose, M. D., and Whitley, L. D. The no free lunch and problem description length.

[84] Shukla, A., Pandey, H. M., and Mehrotra, D. Comparative review of selection techniques in genetic algorithm. In *2015 International Conference on Futuristic Trends on Computational Analysis and Knowledge Management (ABLAZE)* (2015), pp. 515–519.

[85] Singh, G., and Gupta, N. *A Study of Crossover Operators inÂ Genetic Algorithms*. Springer Singapore, Singapore, 2022, pp. 17–32.

[86] Siwei, J., Zhihua, C., Jie, Z., and Yew-Soon, O. Multiobjective optimization by decomposition with pareto-adaptive weight vectors. In *2011 Seventh International Conference on Natural Computation* (2011), vol. 3, pp. 1260–1264.

[87] Syswerda, G. Uniform crossover in genetic algorithms. In *Proceedings of the 3rd International Conference on Genetic Algorithms* (San Francisco, CA, USA, 1989), Morgan Kaufmann Publishers Inc., p. 2–9.

[88] Tan, Y.-Y., Jiao, Y.-C., Li, H., and Wang, X.-K. A modification to moea/d-de for multiobjective optimization problems with complicated pareto sets. *Information Sciences 213* (2012), 14–38.

[89] Thomson, S. L., Ochoa, G., and Vérel, S. Clarifying the difference in local optima network sampling algorithms. In *Evolutionary Computation in Combinatorial Optimization - 19th European Conference, EvoCOP* (2019), vol. 11452 of *Lecture Notes in Computer Science*, Springer, pp. 163–178.

[90] Thomson, S. L., Ochoa, G., Vérel, S., and Veerapen, N. Inferring future landscapes: Sampling the local optima level. *Evol. Comput. 28*, 4 (2020), 621–641.

[91] Trivedi, A., Srinivasan, D., Sanyal, K., and Ghosh, A. A survey of multiobjective evolutionary algorithms based on decomposition. *IEEE TEVC 21*, 3 (2017), 440–462.

[92] Vanneschi, L., Tomassini, M., Pirola, Y., Verel, S., Collard, P., and Mauri, G. A quantitative study of neutrality in gp boolean landscapes. *GECCO 2006 - Genetic and Evolutionary Computation Conference 1* (07 2006), 895–902.

[93] Vanneschi, L., Vérel, S., Tomassini, M., and Collard, P. NK landscapes difficulty and negative slope coefficient: How sampling influences the results. In *Applications of Evolutionary Computing, EvoWorkshops* (2009), vol. 5484 of *Lecture Notes in Computer Science*, Springer, pp. 645–654.

[94] Verel, S., Liefooghe, A., Jourdan, L., and Dhaenens, C. Pareto local optima of multiobjective NK-landscapes with correlated objectives. In *Evolutionary Computation in Combinatorial Optimization*. Springer, 2011, pp. 226–237.

[95] Verel, S., Liefooghe, A., Jourdan, L., and Dhaenens, C. On the structure of multiobjective combinatorial search space: MNK-landscapes with correlated objectives. *European Journal of Operational Research 227*, 2 (2013), 331–342.

[96] Weinberger, E. D. Correlated and uncorrelatated fitness landscapes and how to tell the difference. *Biol Cybern 63*, 5 (1990), 325–336.

[97] Weise, T., and Wu, Z. Difficult features of combinatorial optimization problems and the tunable w-model benchmark problem for simulating them. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion* (New York, NY, USA, 2018), GECCO '18, Association for Computing Machinery, p. 1769–1776.

[98] Wickham, H. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2016.

[99] Wolpert, D., and Macready, W. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation 1*, 1 (1997), 67–82.

[100] Wolpert, D. H., and Macready, W. G. No free lunch theorems for search. Working papers, Santa Fe Institute, 1995.

[101] WRIGHT, S. The roles of mutation, inbreeding, crossbreeding and selection in evolution. *Proceedings of the sixth international congress of Genetics 1* (1932), 356–366.

[102] Xu, Q., Xu, Z., and Ma, T. A survey of multiobjective evolutionary algorithms based on decomposition: Variants, challenges and future directions. *IEEE Access 8* (2020), 41588–41614.

[103] Yadav, S., and Sohal, A. Study of the various selection techniques in genetic algorithms.

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

[105] Zitzler, E., Knowles, J., and Thiele, L. *Quality Assessment of Pareto Set Approximations*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 373–404.

[106] Zitzler, E., Knowles, J., and Thiele, L. *Quality Assessment of Pareto Set Approximations*. Springer-Verlag, Berlin, Heidelberg, 2008, p. 373–404.

[107] Zitzler, E., and Thiele, L. Multiobjective optimization using evolutionary algorithms — a comparative case study. In *Parallel Problem Solving from Nature — PPSN V* (Berlin, Heidelberg, 1998), A. E. Eiben, T. Bäck, M. Schoenauer, and H.-P. Schwefel, Eds., Springer Berlin Heidelberg, pp. 292–301.

[108] Zitzler, E., and Thiele, L. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE Trans. Evol. Comput. 3* (1999), 257–271.

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

[110] Zwillinger, D., and Kokoska, S. *CRC standard probability and statistics tables and formulae*. Boca Raton : Chapman & Hall/CRC, 01 2000.