A Modular Surrogate-assisted Framework for Expensive Multiobjective Combinatorial Optimization

Geoffrey Pruvost, Bilel Derbel, Arnaud Liefooghe, Sébastien Verel, Qingfu Zhang

A Modular Surrogate-assisted Framework for Expensive Multiobjective Combinatorial Optimization

Geoffrey Pruvost, Bilel Derbel, Arnaud Liefooghe, Sébastien Verel, Qingfu Zhang

***Abstract*—The aim of this paper is to push a step towards the development of a surrogate-assisted methodology for expensive optimization problems that have both a combinatorial and a multiobjective nature. We target pseudo-boolean multiobjective functions, and we provide a comprehensive study on the design of a modular framework integrating three main configurable components. The proposed framework is based on the Walsh basis as a surrogate, and on a decomposition-based evolutionary paradigm for maintaining the solution set. The three considered components are: (i) the inner optimizer used for handling the so-constructed Walsh surrogate, (ii) the selection strategy allowing to decide which solution is to be evaluated by the expensive objectives, and (iii) the strategy used to setup the Walsh order hyper-parameter. Based on a thorough empirical analysis relying on two benchmark problems, namely bi-objective NK-landscapes and UBQP problems, we show the effectiveness of the proposed framework with respect to the available budget in terms of calls to the evaluation function. More importantly, our empirical findings shed more lights on the combined effects of the investigated components on search performance, thus providing a better understanding of the key challenges for designing a successful surrogate-assisted multiobjective combinatorial search process.**

***Index Terms*—Multiobjective optimization, discrete surrogates, decomposition.**

I. Introduction

THIS paper investigates the design of efficient solving techniques for tackling complex optimization problems, for which evaluating the quality of one single solution is CPU-time intensive. This is, for instance, the case in different application fields that undergo heavy and costly simulation efforts. More particularly, we target *multiobjective* optimization problems, where a number of objectives are to be optimized simultaneously. Multiobjective optimization is a natural outcome in different application fields — e.g., multi-disciplinary engineering or energy production — which require the decision maker to be provided with a whole set of solutions with different trade-offs among the objectives; i.e., the Pareto set. Therefore, in expensive optimization, one has to deal with the multiobjective nature of the problem at hand by computing a high-quality approximation set, while minimizing the computational effort as much as possible in terms of calls to the expensive evaluation function. In this context, *surrogate-assisted* multiobjective evolutionary algorithms have received intensive research investigations during the last decade [1], [2].

Generally speaking, the basic idea of most surrogate-assisted techniques is to use a meta-model as a substitute of the expensive objective function(s). In single-objective optimization, the meta-model is used to efficiently sample or pre-screen new promising solutions that are then evaluated for real, using the true expensive function. Leveraging such an approach to the multiobjective setting is challenging from different perspectives. Reviewing the whole literature on the subject is out of the scope of this paper. For a detailed discussion, the reader is referred to [3], [4], [5], to cite a few. However, let us comment that two inter-dependent issues are generally addressed by existing approaches: (i) the nature of the meta-model being used as surrogate, and (ii) the class of the multiobjective solving technique used at the core of the underlying optimization process. Interestingly, there exist a number of well-established meta-models dealing with continuous problems, so that the main effort for designing a surrogate-assisted approach for multiobjective continuous optimization is on the articulation of the meta-model with the global multiobjective search procedure. Unfortunately, such an issue has not be addressed in the context of *combinatorial* optimization, since designing efficient discrete surrogates is a relatively new research question, even when dealing with single-objective expensive optimization. In this respect, the main goal of our work is precisely to contribute to the development of novel surrogate-assisted evolutionary computation techniques for expensive multiobjective combinatorial optimization.

Looking at the specialized single-objective optimization literature, one can find a few recent studies investigating surrogate-assisted combinatorial optimization. To our knowledge, there exist four general-purpose discrete meta-models for combinatorial functions, namely, Radial basis function (RBF) [6], Kriging [7], Bayesian [8], and Walsh [9] models. The first three models are adaptations of their well-established counterparts from continuous optimization, whereas the last one is more specifically designed for discrete problems. In fact, it is shown in [9] that a Walsh surrogate is extremely accurate for approximating *any* pseudo-boolean function, that is, for dealing with a priori arbitrary combinatorial optimization problems with binary variables. In this paper, we hence focus our investigations on such an optimization domain and consider to leverage Walsh models when dealing with

G. Pruvost, B. Derbel and A. Liefooghe are with Univ. Lille, CNRS, Centrale Lille, UMR 9189 – CRIStAL, Inria Lille–Nord Europe, F-59000 Lille, France (e-mails: geoffrey.pruvost@univ-lille.fr, bilel.derbel@univ-lille.fr, arnaud.liefooghe@univ-lille.fr).

S. Verel is with Univ. Littoral Côte d'Opale, LISIC, F-62100 Calais, France (e-mail: verel@lisic.univ-littoral.fr).

Q Zhang is with City Univ. Hong Kong, Kowloon Tong, Hong Kong (e-mail: qingfu.zhang@cityu.edu.hk).

multiobjective pseudo-boolean problems.

To the best of our knowledge, the only existing study on the subject can be found in our recent work [10], which constitutes the first attempt to leverage a single-objective discrete surrogate for multiobjective optimization. In particular, the main goal in [10] was to show that (single-objective) Walsh surrogates [9], [11] can be integrated within the multiobjective evolutionary algorithm based on decomposition MOEA/D [12]. The search process generates a set of offspring solutions on the basis of the Walsh surrogate, considered as a substitute of the true objectives, and then selects one offspring to be evaluated using the real objectives. In this paper, we pursue our preliminary investigations by proposing a number of novel design components and strategies, that allow us to provide substantial improvements in terms search performance. More importantly, the work presented hereafter should *not* be viewed as a hoarse-racing against previous works, or for solving a given optimization problem. Instead, we aim at gaining a more fundamental understanding of what makes a surrogate-assisted approach effective. For this purpose, and besides proposing new design components, we provide a comprehensive and systematic study on the combined effect of the different key design choices described within a general-purpose surrogate-assisted multiobjective combinatorial optimization (S-MCO) framework. The considered framework is thought to be as modular as possible, thus allowing the different design components to be instantiated differently in a plug-and-play fashion. More precisely, three main design components are investigated, as briefly discussed below:

- Using a Walsh model as a substitute for the objectives requires to be able to optimize it accurately. The first design component is hence the inner optimizer used for producing a set of promising solutions by (temporarily) relying on the previously-trained Walsh surrogate. We consider three different classes of optimizers, namely MOEA/D [12], Pareto Local Search (PLS) [13], and an independent multiple local search (MLS) [14]. Our findings suggest that local search engines are to be preferred for optimizing the underlying discrete Walsh surrogate.
- Having a set of candidate solutions obtained from the Walsh surrogate and the inner optimizer, the second component is to decide which solution to select for the true evaluation. This step is critical since the objectives are assumed to be expensive, and one has to accurately select the solution that is believed to be the most beneficial for the whole multiobjective search process. We adopt a decomposition-based approach, where the current approximation set is structured using scalarizing weighted functions as in MOEA/D. We consider four different decomposition-based strategies, where each candidate solution is given a score computed on the basis of their (predicted) scalar values and the current approximation set. We show that the effect of this component on search performance depends on the nature of the inner optimizer, and that a score rendering the expected improvement in terms of the scalar value is to be preferred.
- Last but not least, as any meta-model, the Walsh surrogate comes with a critical hyper-parameter, corresponding to the order of the underlying Walsh basis used for fitting the meta-model. From a theoretical perspective, every pseudo-boolean function has an exact Walsh decomposition up to some order, which depends on the level of interaction among the variables, and which can be computed assuming that an unlimited budget is provided. However, when used as a surrogate, the Walsh model has to be trained based on a relatively restricted sample with respect to a possibly black-box (theoretically unknown) problem. Consequently, the third component deals with the setting of the order to be used when constructing the Walsh surrogate. We compare three simple strategies where the order is either (i) static, or is evolving following two basic dynamic strategies, namely (ii) in a random manner, or (iii) in a greedy manner, according to the current state of the search process. We also found that a dynamic strategy provides very competitive results, especially when the exact Walsh order is unknown.

The remainder of the paper is organized as follows. In Section II, we recall basic definitions for multiobjective optimization, and we briefly introduce the background for multiobjective decomposition and discrete Walsh surrogates. In Section III, we describe the different design components of the S-MCO framework. In Section IV, we give our experimental setup. In Section V, we report our empirical findings. In Section VI, we conclude the paper and we discuss further research directions.

II. Background and Preliminaries

A. Multiobjective Combinatorial Optimization

We assume that we are given a black-box objective vector function $F = (f_1, f_2, \ldots, f_m)$, such that each objective f_i, $i \in [\![1, m]\!]$, is to be maximized, and a set X of solutions in the *variable space*. When X is a discrete set, we face a *multiobjective combinatorial optimization problem* (MCOP). In particular, we consider unconstrained pseudo-boolean multiobjective optimization problems, such that $F\colon \{0,1\}^n \mapsto \mathbb{R}^m$, where n is the problem size. Let $Z = f(X) \subseteq \mathbb{R}^m$ be the set of feasible outcome vectors in the *objective space*. An objective vector $z \in Z$ is *dominated* by a vector $z' \in Z$ iff $\forall i \in [\![1, m]\!]$, $z_i \leqslant z'_i$, and $\exists j \in [\![1, m]\!]$ s.t. $z_j < z'_j$. A solution $x \in X$ is dominated by a solution $x' \in X$ iff $F(x)$ is dominated by $F(x')$. A solution is *Pareto optimal*, or *non-dominated*, if there does not exist any other solution that dominates it. The set of all Pareto optimal solutions is the *Pareto set*. Its mapping in the objective space is the *Pareto front*. Identifying the Pareto set is known to be an NP-hard task for a wide range of MCOPs, and the Pareto set typically contains an exponential number of solutions [15]. As such, we often have to rely on a Pareto set *approximation*, for which a large number of multiobjective evolutionary algorithms have been proposed since the early nineties [16], [17].

B. Decomposition-based Multiobjective Optimization

In the broad range of multiobjective evolutionary algorithms, approaches based on *decomposition* are amongst the

state-of-art [18]. In particular, MOEA/D [12] decomposes the multiobjective optimization problem into a set of single-objective sub-problems that target different regions of the Pareto front. The population of solutions $\mathcal{P}_\mu = \{x^1, \ldots, x^\mu\}$ is evolved such that each solution x^i, $i \in [\![1, \mu]\!]$, is assigned to a weight vector ω^i corresponding to a given sub-problem. A sub-problem then seeks for a high-quality solution with respect to an aggregation function $\mathsf{g}(x \mid \omega^i, F)$, parameterized by the weight vector ω^i. The population $\mathcal{P}_\mu$ is evolved following conventional evolutionary mechanisms, such as selection and variation, in order to optimize the different sub-problems.

Given a weight vector ω^i, the aggregation function transforms an objective vector into a scalar value. One recommended function that we consider in this paper is the Chebyshev function, to be minimized, which is defined as follows:

$$\mathsf{g}(x \mid \omega, F) = \max_{j \in [\![1,m]\!]} \omega_j \cdot \left| z^\star_j - f_j(x) \right| \tag{1}$$

where $x \in X$, $\omega = (\omega_1, \ldots, \omega_m)$ is a positive weight vector, and $z^\star$ is a reference point such that $z^\star_j > f_j(x), \forall x \in X, j \in [\![1, m]\!]$.

One of the most distinguishable feature of MOEA/D is that the population is evolved simultaneously and cooperatively. A solution that is currently assigned to a given sub-problem can become parent to an offspring generated for another sub-problem, and vice versa. A newly-generated offspring is compared to solutions assigned to others sub-problems by means of their corresponding aggregation function. At a given generation, offspring can replace multiple solutions from the population, assuming that they improve multiple sub-problems. In the original MOEA/D, this cooperation is limited by a *neighborhood*, such that an offspring cannot be generated by parents outside this neighborhood, neither it can replace solutions outside its neighborhood. In our work, we consider that the parent selection and replacement mechanisms are *not* limited by such a neighborhood, and the whole population is used instead. This design choice is motivated by two observations: (i) recent studies reveal that this can actually improve the performance of MOEA/D for MCOPs [19], and (ii) we aim at accelerating the convergence of the population to the Pareto front, because of the particularly limited budget induced in expensive optimization.

C. Walsh Surrogates

Even in single-objective optimization, surrogate models for black-box *combinatorial* functions are scarce [20]. Furthermore, to our knowledge, surrogate models have never been applied to *expensive* MCOPs — with the exception of our preliminary study [10]. Recently, a surrogate model was proposed for pseudo-boolean functions [9], and has been successfully applied in expensive single-objective optimization [11], showing its superiority against previous proposals [21], [6], [7], [8]. This model is based on Walsh functions and is described below.

1) Walsh Basis: Walsh functions [22] constitute an enumerable set of functions $\phi_\ell : [0, 1] \to \{-1, 1\}$ which composes a normal and orthogonal basis of the Hilbert space $L^2([0, 1])$. Like the trigonometric functions of the Fourrier basis, they can be used to decompose any function from the Hilbert space under some mild conditions [22], and have been used since the late seventies in the theory of evolutionary computation [23].

Given a pseudo-boolean function $f\colon \{0,1\}^n \mapsto \mathbb{R}$, Walsh functions are defined as follows [9]. For any integer $\ell \in [\![0, 2^n - 1]\!]$, following the binary representation $\ell = \sum_i \ell_i 2^i$ with $\ell_i \in \{0, 1\}$, the Walsh function $\phi_\ell : \{0,1\}^n \to \{-1, 1\}$ is defined for any binary string $x = (x_1, \ldots, x_i, \ldots, x_n) \in \{0,1\}^n$ as follows:

$$\phi_\ell(x) = (-1)^{\sum_{i=0}^{n-1} \ell_i x_i} \tag{2}$$

The *order* of a Walsh function ϕ_ℓ, denoted by $o(\phi_\ell)$, is defined by the number of binary digits equals to 1 in the binary representation of ℓ. For example, the Walsh function of order 0 is ϕ_0, the Walsh functions of order 1 are ϕ_{2^p} for all integers $p \geqslant 0$, the Walsh functions of order 2 are $\phi_{2^p+2^{p'}}$ for all pairs of integers $p \neq p' \geqslant 0$, and so on.

2) Exact Walsh Transform: The so-defined (finite) set of discrete functions is a normal orthogonal basis for the space of pseudo-boolean functions, i.e., $\forall \ell, \ell' \in [\![0, 2^n - 1]\!]$, $\frac{1}{2^n} \sum_{x \in \{0,1\}^n} \phi_\ell(x) \cdot \phi_{\ell'}(x) = \delta_{\ell\ell'}$. As such, any pseudo-boolean function f can be written as:

$$f(x) = \sum_{\ell=0}^{2^n-1} w_\ell \cdot \phi_\ell(x) \tag{3}$$

$$w_\ell = \frac{1}{2^n} \sum_{x \in \{0,1\}^n} f(x) \cdot \phi_\ell(x) \tag{4}$$

which is to recall the Fourier transform.

3) Approximated Walsh Decomposition: On one hand, the Walsh functions ϕ_ℓ are uniquely defined and do not depend on the problem at hand; see Eq. (2). On the other hand, the values of the coefficients w_ℓ do depend on the considered function f, as given in Eq. (4). On top of that, there might exist an exponential number of non-zero coefficients in the exact Walsh transform as given in Eq. (3). Roughly speaking, the coefficients corresponding to some Walsh function of a given order o capture the interaction among a set of o variables in function f; i.e., they render how the function value is affected when the variables change. This means that, unless a large number of variables interact with each other, many coefficients are expected to have a zero value. This is the case for multiple combinatorial problems with a quadratic or cubic number of variable interactions, and hence a reasonable assumption in practice for a wide range of problem classes. Hence, the idea developed in [9], [11] is to approximate f using solely the Walsh functions up to a (small) constant order $d \ll n$, and to use an estimate $\widehat{w}_\ell$ of the (unknown) coefficient w_ℓ. More formally, given a *constant* order d, a function f can be approximated by the following model:

$$\tilde{f}(x \mid d) = \sum_{\ell \,:\, o(\phi_\ell) \leqslant d} \widetilde{w}_\ell \cdot \phi_\ell(x) \tag{5}$$

For example, a second-order approximation can be rewritten as:

$$\tilde{f}(x \mid 2) = \widetilde{w}_0 + \sum_{i=1}^{n} \widetilde{w}_i \cdot (-1)^{x_i} + \sum_{i<j<n} \widetilde{w}_{ij} \cdot (-1)^{x_i + x_j}$$

where $\widetilde{w}_0$ is the zero-order estimated coefficient, and $\widetilde{w}_i$ and $\widetilde{w}_{ij}$ are first- and second-order estimated coefficients, respectively. Intuitively, the larger the order d, the more accurate the expansion to approximate the original function.

4) Walsh Surrogate Model: Constructing a discrete Walsh surrogate up to constant order d consists in computing the approximate coefficients $\widetilde{w}_\ell$. This can be done by fitting the approximated model and estimating the value of $\widetilde{w}_\ell$ as in standard supervised machine learning approaches. More specifically, this turns out to be a standard linear regression problem, since Eq. (5) can be interpreted as a linear model whose predictors are the Walsh function values. In particular, sparse techniques can be used to minimize the number of non-zero coefficients when the number of predictors is large [24]. Following [9], [11], we hence use the Lasso algorithm [25] to fit the model, which is of particular interest given that the number of Walsh functions of up to a given order d might be greater than the number of solutions used for training.

In the following, we consider a modular integration of Walsh surrogates into decomposition-based multiobjective approaches for solving expensive MCOPs, and we discuss the setting of the Walsh order d, considered as a hyper-parameter of the meta-model.

III. A Surrogate-assisted Framework for Expensive MCOP

Following our preliminary work [10], we provide a detailed step-by-step description of a surrogate-assisted multiobjective combinatorial optimization (S-MCO) modular framework. We subsequently discuss the three main proposed design components, namely: (i) the inner optimizer of the substitute Walsh surrogate, (ii) the selection of the solution to be evaluated at each iteration, and (iii) the setting of the Walsh order at the training phase.

A. General Description

The high-level pseudo-code of the proposed S-MCO framework is depicted in the template of Algorithm 1. The S-MCO framework follows the general computational flow of the (surrogate-less) MOEA/D, as described in Section II-B, with a few exceptions. The MCOP is decomposed into μ sub-problems, for which one seeks a high-quality solution. A population $\mathcal{P}_\mu$ is initialized with μ randomly-generated solutions. Each solution is evaluated using the expensive objectives before being assigned to a unique sub-problem. This population is also used as the initial dataset $\mathcal{D}$ for training the surrogates. The algorithm iterates over all sub-problems to perform one generation, and subsequent generations are performed until the budget — in terms of calls to the expensive objective vector function — is exhausted.

For a given iteration dealing with a sub-problem $i \in [\![1, \mu]\!]$, the first step consists in constructing the surrogates with the order currently chosen by the Walsh order selection component (line 7), which to be discussed in more details in Section III-D. Notice that the Walsh order component takes as input a *History* variable, which is an artifact indicating that this component may use some information about the search status for deciding which order to return. For each objective function f_i in $F = (f_1, \ldots, f_i, \ldots, f_m)$, we then consider one surrogate $\widetilde{f}_i$, hence ending up with m surrogates. These surrogates $\widetilde{\mathcal{F}} = (\widetilde{f}_1, \ldots, \widetilde{f}_m)$ are trained (line 8) with the same Walsh order $\mathcal{O}$ and with *all* solutions evaluated so far, stored into the (training) dataset $\mathcal{D}$. The corresponding estimate of the Walsh coefficients (Eq. (5)) for each surrogate $\widetilde{f}_i$ are computed following a sparse linear regression methodology, as described in Section II-C.

In contrast with the conventional MOEA/D, S-MCO does not generate any offspring or candidate solution right away by using standard variation operators, such as mutation or crossover. Instead, it temporarily relies on the surrogates, and intensively searches for high-quality solutions. Only after,

Algorithm 1: The surrogate-assisted framework for multiobjective combinatorial optimization (S-MCO)

Input: $\mathcal{W}_\mu := \{\omega^1, \ldots, \omega^\mu\}$: weight vectors; $\mathsf{g}(\cdot \mid \omega, \cdot)$: a scalarizing function to be minimized; d: maximum order of Walsh functions;

1 $\mathcal{P}_\mu \leftarrow \{x^1, \ldots, x^\mu\}$: initial population of size μ;
2 $\mathcal{D} \leftarrow \{(x^1, F(x^1)), \ldots, (x^\mu, F(x^\mu))\}$: training data ;
3 $\mathcal{EP} \leftarrow$ initialize external archive (optional) ;
4 $z^\star \leftarrow$ initialize reference point;
5 **while** *global budget is not exhausted* **do**
6 **for** $i \in \{1, \ldots, \mu\}$ **do**
  /* Choose the Walsh order */
7 $\mathcal{O} \leftarrow$ ChooseWalshOrder$(History, d)$;
  /* Train Walsh models */
8 $\widetilde{\mathcal{F}} := (\widetilde{f}_1, \ldots, \widetilde{f}_m) \leftarrow$ TrainWalshModels$(\mathcal{D}, \mathcal{O})$;
  /* Copy the reference point to use it in the optimizer and the selection strategy */
9 $z^{\star\star} \leftarrow z^\star$;
  /* Run the optimizer with the surrogate model */
10 $\mathcal{S} \leftarrow$ RunOptimizer$(\mathcal{P}_\mu, \widetilde{\mathcal{F}}, z^{\star\star})$;
  /* Select a solution for true evaluation */
11 $x' \leftarrow$ SelectForEvaluation$(\mathcal{S}, \omega_i, \widetilde{\mathcal{F}}, z^{\star\star})$;
12 $F(x') \leftarrow$ evaluate x' ;
13 $\mathcal{EP} \leftarrow$ (optional) update external archive using x' ;
14 $z^\star \leftarrow$ update reference point using $F(x')$;
  /* Replacement process in the population */
15 **for** $j \in \{1, \ldots, \mu\}$ **do**
16 **if** $\mathsf{g}(x' | \omega^j, F) < \mathsf{g}(x^j | \omega^j, F)$ **then**
17 $x^j \leftarrow x'$;
  /* Update training data */
18 $\mathcal{D} \leftarrow \mathcal{D} \cup \{(x', F(x'))\}$;

it selects one of them without performing any call to the expensive objectives. In other words, after the training step, a pool of solutions $\mathcal{S}$ is generated by the *surrogate optimizer* (line 10), which constitutes our second component. Solutions from $\mathcal{S}$ have not been evaluated with the true objectives F, but they have been estimated with the surrogates $\widetilde{\mathcal{F}}$. This component takes the current population as input, together with the current surrogates $\widetilde{\mathcal{F}}$ and a copy of the reference point $z^{\star\star}$. It is important to use a copy of $z^{\star}$ in the surrogate optimizer because the reference point will likely be updated during the process, but only with (unreliable) estimated objective values, whereas $z^{\star}$ is only updated according to the true objective values evaluated so far. The different design choices for this inner surrogate optimization component are discussed in details in Section III-B. The next step (line 11) selects the most interesting candidate solution from $\mathcal{S}$, to be evaluated using the expensive objectives F. We call this component the *selection strategy*, and we describe it in Section III-C. It takes the pool of solutions $\mathcal{S}$ generated by the optimizer as input, together with the surrogate model $\widetilde{\mathcal{F}}$, the reference point $z^{\star\star}$ as updated by the optimizer, and the current weight vector ω_i. As output, it returns the solution to be evaluated next at the current iteration i. Notice that the weight vector ω_i given as input of the selection component is only an artifact of our template in order to indicate that this component may, or not, depend on the current iteration i. In fact, the different iterations of S-MCO, as implemented within the for loop in line 6, are designed in the same style as MOEA/D, and might be thought with respect to the different sub-problems. However, as it will be discussed later, the selection strategy can be designed in such a way that the solution being selected is believed to be the best for the overall search progress, and not necessarily for the particular sub-problem i.

After the selection and the (true) evaluation of the solution x', we follow the standard process of MOEA/D, with the update of the external archive $\mathcal{EP}$, the update of the reference point $z^{\star}$ and the replacement of the population. At this step, we compare the aggregation value of the candidate solution x' with *all* solutions from the current population $\mathcal{P}_\mu$, according to their corresponding weight vectors. The candidate solution x' replaces solutions from the population wherever there is an improvement. In other words, we do not use a fixed-size neighborhood for replacement, as performed in the conventional MOEA/D. We rather adopt an elitist replacement strategy involving the whole population, which is clearly to be attributed to the particularly restricted budget we are assuming. The last step is to update the dataset $\mathcal{D}$ by adding the new evaluated solution x' (line 18).

The three generic components of the S-MCO framework can be configured in different manners. Different proposed design choices are discussed in details below.

B. Component #1: Surrogate Optimizer

In [10], we highlighted the benefit of using a multiobjective optimizer to search for good-quality solutions with respect to the surrogates. This was a clear improvement over using variation operators to blindly produce offspring, and pre-screening them. It is important to notice that the surrogate optimizer does not evaluate any solution using the true expensive objectives F, but uses the surrogates $\widetilde{\mathcal{F}}$ to quickly estimate their quality. Assuming that the goodness of fit of the surrogates is high, the surrogate optimizer allows us to intensively search for good-quality solutions in a cheap way. In this paper, we consider three different surrogate optimizers (line 10 in Algorithm 1), and we analyze the impact of this component on the overall performance of the S-MCO framework:

- **$\mathsf{Optim}_{\mathsf{MOEA/D}}$** performs the conventional MOEA/D for a number of generations in order to identify a Pareto set approximation for the surrogates $\widetilde{\mathcal{F}}$. $\mathsf{Optim}_{\mathsf{MOEA/D}}$ is given the current population from the main S-MCO algorithm, and returns the evolved population seeking to optimize $\widetilde{\mathcal{F}}$. This strategy is referred as the substitution strategy in [10] and is actually the only optimizer that was investigated therein.
- **$\mathsf{Optim}_{\mathsf{MLS}}$** (Algorithm 2) independently runs multiple local search, one for each sub-problem defined in S-MCO. More precisely, given the surrogates $\widetilde{\mathcal{F}}$, for each single-objective sub-problem defined by the weight vector ω^i, $i \in [\![1, \mu]\!]$, a standard hill-climber is performed to compute a good-quality solution. Each hill-climber is initialized with a random solution, and iteratively selects an improving solution using a standard 1-bit-flip neighborhood, until the search stops in a local optima with respect to the aggregation function $\mathsf{g}(\cdot \mid \omega^i, \widetilde{\mathcal{F}})$. $\mathsf{Optim}_{\mathsf{MLS}}$ then returns a pool of μ (local optimal) solutions, one per sub-problem.
- **$\mathsf{Optim}_{\mathsf{PLS}}$** (Algorithm 3) is based on Pareto Local Search [13]. It maintains an unbounded archive of mutually non-dominated solutions, initialized with a random solution. At each step, one solution is selected at random from the archive, all its neighbors are evaluated with respect to $\widetilde{\mathcal{F}}$, and are used to update the archive. The current solution is then tagged as visited in order to avoid a useless re-exploration of its neighborhood. The search process stops once all solutions from the archive are tagged as visited. The content of the archive corresponds to the pool of solutions returned by $\mathsf{Optim}_{\mathsf{PLS}}$.

C. Component #2: Selection Strategy

The selection of the solution to be expensively evaluated from the whole pool returned by the surrogate optimizer (line 11 in Algorithm 1) is another important component of the S-MCO framework. At this stage, let us recall that the algorithm is at some iteration $i \in [\![1, \mu]\!]$, which can be thought as corresponding to a particular sub-problem ω^i, and that it needs to decide which solution is the most beneficial to be evaluated using the real (expensive) objectives. An intuitive strategy is to attempt to improve sub-problems in a round-robin manner, and thus to select the solution whose estimated scalar value (computed on the basis of the objective values predicted by the surrogate) is the best for the current sub-problem. Such a choice could also be motivated from an exploration perspective, since it allows the search effort to be evenly distributed among the sub-problems. This was actually proposed in our

Algorithm 2: Multiple Local Search (MLS)

Input: $\mathcal{W}_\mu := \{w^1, \ldots, w^\mu\}$: vectors of weights;
$\widetilde{\mathcal{F}} := (\widetilde{f}_1, \ldots, \widetilde{f}_m)$: estimation of the function $\mathcal{F}$;
$\mathsf{g}(\cdot \mid \omega, \cdot)$: a scalarizing function to be minimized;
$\mathcal{N} : X \leftarrow 2^X$: a neighborhood relation;

$\mathcal{S} \leftarrow \varnothing$;
for $\omega^i \in \mathcal{W}_\mu$ **do**
  LocalOptimum $\leftarrow$ False;
  $x^* \leftarrow$ random (initial) solution;
  while *! LocalOptimum* **do**
    $x'^* \leftarrow x^*$;
    foreach $x' \in \mathcal{N}(x^*)$ **do**
      if $\mathsf{g}(x'|\omega^i, \widetilde{\mathcal{F}}) < \mathsf{g}(x'^*|\omega^i, \widetilde{\mathcal{F}})$ **then**
        $x'^* \leftarrow x'$; ;
    if $\mathsf{g}(x'^*|\omega^i, \widetilde{\mathcal{F}}) < \mathsf{g}(x^*|\omega^i, \widetilde{\mathcal{F}})$ **then**
      $x^* \leftarrow x'^*$;
    else
      LocalOptimum $\leftarrow$ True;
  $\mathcal{S} \leftarrow \mathcal{S} \cup \{x^*\}$;
return $\mathcal{S}$

Algorithm 3: Pareto Local Search (PLS)

Input: $\widetilde{\mathcal{F}} := (\widetilde{f}_1, \ldots, \widetilde{f}_m)$: estimation of the function $\mathcal{F}$;
$\mathcal{N} : X \leftarrow 2^X$: a neighborhood relation;

$x \leftarrow$ random (intial) solution;
$\mathcal{S} \leftarrow \{x\}$; `// Archive of non dominated solutions`
$R \leftarrow \{x\}$; `// Remaining non visited solutions`
while $R \neq \varnothing$ **do**
  $x' \leftarrow$ select a solution at random in R;
  foreach $x'' \in \mathcal{N}(x')$ **do**
    if x'' *is <u>not</u> dominated by any solution in* $\mathcal{S}$ **then**
      for $x \in \mathcal{S}$ **do**
        if x *is dominated by* x'' **then**
          $\mathcal{S} \leftarrow \mathcal{S} \setminus \{x\}$;
      $\mathcal{S} \leftarrow \mathcal{S} \cup \{x''\}$;
      for $x \in R$ **do**
        if x *is dominated by* x'' **then**
          $R \leftarrow R \setminus \{x\}$;
      $R \leftarrow R \cup \{x''\}$;
  $R \leftarrow R \setminus \{x'\}$;
return $\mathcal{S}$

preliminary work [10]. However, it remains unclear if such a design choice is optimal for expensive MCOPs, since the budget is very restrictive and more aggressive strategies could potentially lead to a better convergence behavior. In this paper, we hence investigate four different selection strategies, based on the (predicted) aggregation value or the improvement of candidate solutions.

More specifically, for an iteration $i \in [\![i, \mu]\!]$, we consider the following four selection strategies:

Select$_{\text{local}}$ chooses the solution x' with the best aggregation value for the sub-problem considered at the current iteration and correspondingly to the weight vector ω^i.

$$x' := \text{argmin}_{x \in \mathcal{S}}\, \mathsf{g}(x \mid \omega^i, \widetilde{\mathcal{F}})$$

Select$_{\text{global}}$ chooses the solution x' with the best aggregation value with respect to any sub-problem $\ell \in [\![1, \mu]\!]$, independently of the current iteration:

$$x' := \text{argmin}_{x \in \mathcal{S}} \min_{1 \leq \ell \leq \mu} \mathsf{g}(x \mid \omega^\ell, \widetilde{\mathcal{F}})$$

Select$_{\text{BI}}$ (best improvement) chooses the solution x' that improves the most the aggregation value of any solution from the population, independently of the current iteration:

$$x' := \text{argmin}_{x \in \mathcal{S}} \min_{1 \leq \ell \leq \mu} \mathsf{g}(x_\ell \mid \omega^\ell, \widetilde{\mathcal{F}}) - \mathsf{g}(x \mid \omega^\ell, \widetilde{\mathcal{F}})$$

Select$_{\text{BI}_{\text{norm}}}$ (best improvement, normalized) is based on the previous strategy, but the improvement value is normalized by the actual aggregation value of the current solution associated with each sub-problem. The selected offspring x' is then:

$$x' := \text{argmin}_{x \in \mathcal{S}} \min_{1 \leq \ell \leq \mu} \frac{\mathsf{g}(x_\ell \mid \omega^\ell, \widetilde{\mathcal{F}}) - \mathsf{g}(x \mid \omega^\ell, \widetilde{\mathcal{F}})}{\mathsf{g}(x_\ell \mid \omega^\ell, \widetilde{\mathcal{F}})}$$

It is important to notice that, in order to determine the quality of a solution, the selection strategy uses the surrogates $\widetilde{\mathcal{F}}$ and not the expensive objectives F. The (real) evaluation function F is only used at line 12 in Algorithm 1, after the selection strategy returns the candidate solution chosen for the considered iteration.

D. Component #3: Walsh Order

As pointed out in Section II-C, the Walsh surrogate requires an order up to which the coefficients are expanded, which is the only hyper-parameter to set prior to fitting. The choice of the Walsh order is an important and difficult task for black-box problems. This choice dictates the number of coefficients in the surrogate model (Eq.(5)). This does not only impact the theoretical model accuracy, depending on the problem being solved. It also impacts the fitting process itself, because the model complexity increases with the number of coefficients. Let us illustrate this with a problem of size $n = 50$. In such a case, there are 51 coefficients for a Walsh order $d = 1$, $1\,276$ coefficients for an order $d = 2$, and $20\,876$ coefficients for an order $d = 3$. A high number of coefficients eventually improves the approximation of the actual function, but then a larger dataset is expected in order to accurately train the model, which can be an issue since we can only assume a restricted size for the training dataset. By contrast, a low number of

coefficients can be insufficient to accurately approximate the actual objectives.

In the following, we assume that the order used to effectively train the Walsh model is bounded by some maximum constant value $d_{\max}$. This is a reasonable assumption in practice, since otherwise the number of Walsh coefficients to estimate would grow exponentially, and the cost of fitting would not be reasonable. We also remark that such an assumption is in line with the fact that many challenging combinatorial optimization problems can be decomposed in an exact manner using Walsh functions of bounded order. However, it remains unclear which concrete order to consider within a given range $[\![1, d_{\max}]\!]$, and whether this order shall remain static or if its optimal value changes during the curse of the search process when integrated into a surrogate-assisted approach. Consequently, we investigate three simple strategies for setting the Walsh order, as described below.

Order$_{\text{static}}$ corresponds to a parameterized strategy for setting the Walsh order. The order is then a static parameter of the S-MCO framework, and its value is chosen for the whole search process, i.e., the Walsh surrogate is fitted considering a fixed order value $d \in [\![1, d_{\max}]\!]$, where d is assumed to be a parameter provided by the end-user.

Order$_{\text{random}}$ is a basic dynamic strategy where the order is set randomly every time the model is trained. In other words, at each iteration, the value of the Walsh order is picked uniformly at random within the range $[\![1, d_{\max}]\!]$. The advantage of this strategy is to use all orders without being completely limited by an improper static setting. This random choice will also serve as a baseline when measuring the impact of this component in our empirical investigations.

Order$_{\text{greedy}}$ dynamically adjusts the Walsh order during the search process, depending on the data collected so far. At the end of each iteration $i \in [\![1, \mu]\!]$, we store how many sub-problems were improved (in line 17) after evaluating the offspring selected for true evaluation. Let us denote by p_j the number of improved sub-problems, where j corresponds to the j^{th} true function evaluation. We thereby track the number of improved sub-problems over a window of t (last) iterations, where t is a user-defined parameter. At the beginning of each iteration, we then compute the average, over the window t, of the number of improved sub-problems, denoted $\overline{p}_t$. If $\overline{p}_t < 1$, then the strategy consists in changing the Walsh order according to a particular schedule, as discussed in the following. Let d_{curr} be the Walsh order used at some current iteration $i \in [\![1, \mu]\!]$. Let also d_{prev} be the Walsh order used at the previous iteration (i.e., iteration $i-1$ or μ). Now let us assume, at the beginning of the next iteration, that we have $\overline{p}_t < 1$. Then, the new order d_{new} returned by the selection strategy is given by $d_{\text{new}} = d_{\text{curr}}+1$ if either $d_{\text{curr}} = 1$ or $d_{\text{prev}} < d_{\text{curr}} < d_{\max}$; otherwise, we set $d_{\text{new}} = d_{\text{curr}} - 1$.
The idea behind this greedy dynamic strategy is to start by using a small Walsh order, and to observe whether any improvement is obtained in the population. In this case, we keep using the same successful order. Otherwise, we increase the order, and hence the model complexity, in the hope of obtaining a better fitted surrogate, thus allowing the search process to identify an improving offspring. However, when attaining the maximum allowed complexity for the model $d_{\max}$, we continue scanning for possible orders by decreasing the current value, and increasing it again if an order of 1 is reached. The order value stops changing when improvement are found, and the whole dynamic process is repeated.

IV. Experimental Setup

In the remainder of the paper, we investigate the impact of the proposed components on the performance of the considered S-MCO framework. Obviously, extensive experiments with real-world expensive black-box MCOPs would be computationally infeasible. We also recall that our main goal is not to tackle a particular problem, but to conduct a general-purpose study that aims at providing guidelines for designing an effective surrogate-assisted approach. For this purpose, we rather consider two different classes of well-established MCOP benchmarks. They are described below, followed by a summary of the experimented algorithm variants.

A. Benchmark Problems

We consider bi-objective NK-landscapes [26], [27] and bi-objective unconstrained binary quadratic programming problems (UBQP) [28], as a set of challenging and representative pseudo-boolean MCOPs. In fact, NK-landscapes allows one to control the degree of non-linearity of the problem, we hence consider instances with a variable degree of difficulty. UBQP is a computationally challenging problem which is known to embrace a remarkable range of applications in combinatorial optimization [29].

1) Multiobjective NK-Landscapes (MNK-Landscapes).: Solutions are binary strings of size n and the objective vector, to be maximized, is defined as $F\colon \{0,1\}^n \mapsto [0,1]^m$. An objective value $f_i(x)$ of a solution $x = (x_1, \ldots, x_n)$ is given by [30]:

$$f_i(x) = \frac{1}{n}\sum_{j=1}^{n} c_j^i(x_j, x_{j_1}, \cdots, x_{j_k}) \tag{6}$$

where the $c_j^i : \{0,1\}^{k+1} \to [0,1)$ are component functions, and k is a parameter specifying the number of epistatic interactions. The component function c_j^i assigns a real-valued contribution for every combination of x_j and its k epistatic interactions $\{x_{j_1}, \ldots, x_{j_k}\}$. The parameter k defines the degree of non-linearity of the problem, and hence the *ruggedness* of the landscape.

We consider bi-objective MNK-landscapes [27] with the following setting: $n \in \{25, 50\}$ and $k \in \{0, 1, 2\}$. In line with previous studies [10], [11], this allows us to span instances ranging from a small to a relatively large number of variables, and with a linear ($k = 0$), to quadratic ($k = 1$) and cubic ($k = 2$) number of interactions.

Table I
S-MCO COMPONENTS AND VALUES INVESTIGATED IN THE EXPERIMENTAL ANALYSIS.

component	values	analyzed in
Surrogate optimizer	$\{\mathsf{Optim}_{\mathsf{MOEA/D}}, \mathsf{Optim}_{\mathsf{MLS}}, \mathsf{Optim}_{\mathsf{PLS}}\}$	Section V-B
Selection strategy	$\{\mathsf{Select}_{\mathsf{local}}, \mathsf{Select}_{\mathsf{global}}, \mathsf{Select}_{\mathsf{BI}}, \mathsf{Select}_{\mathsf{BI}_{\mathsf{norm}}}\}$	Section V-A
Walsh order	$\{\mathsf{Order}_{\mathsf{static}}, \mathsf{Order}_{\mathsf{random}}, \mathsf{Order}_{\mathsf{greedy}}\}$ with $d \in \{1, 2, 3\}$	Section V-C

2) Multiobjective UBQP (MUBQP): Given a collection of n items such that each pair of items is associated with multidimensional profit values, the multiobjective unconstrained binary quadratic programming (MUBQP) problem seeks a subset of items that maximizes the sum of their paired values in each dimension [28]. The value of a pair is summed up only if the two corresponding items are selected. A solution can be represented as a binary string of size n. Each position from the binary string maps to a particular variable that indicates whether the corresponding item is included in the subset of selected items or not. Given a solution $x = (x_1, \ldots, x_n)$, each objective f_k, $k \in [\![1, , m]\!]$, is defined as follows:

$$f_k(x) = \sum_{i=1}^{n} \sum_{j=1}^{n} q_{ij}^k x_i x_j \tag{7}$$

where $Q_k = q_k^{ij}$ is an $n \times n$ matrix of constant values, either positive, negative or zero, $k \in [\![1, m]\!]$. We consider bi-objective UBQP instances with $n \in \{25, 50\}$ variables and a density of 0.9 for non-zero entries in the matrix Q_k.

B. Algorithms and Parameter Setting

In order to fully examine the different components introduced in the proposed S-MCO framework, we systematically evaluate the performance of *all* its possible combinations. Let us notice that the recommended surrogate algorithm from our preliminary investigations [10] corresponds to the S-MCO framework configured with the $\mathsf{Optim}_{\mathsf{MOEA/D}}$ and the $\mathsf{Select}_{\mathsf{local}}$ strategies. We hence shall keep this as a baseline allowing to fully appreciate our findings. In fact, notice that besides being able to substantially improve the search performance compared to [10], our primary goal is to conduct a comprehensive analysis showing the impact of the different components on search performance, and to highlight the key challenges for an effective surrogate-assisted multiobjective combinatorial optimization search. Furthermore, and since we consider three surrogate optimizers in our study, we shall analyze the search performance obtained with each optimizer when used as a main optimization procedure for the original expensive problem, independently of the S-MCO framework. This allows us to analyze the relative benefits of using a surrogate-assisted approach.

For all S-MCO configurations, we use 50 weight vectors such that $\omega^i = \left(\frac{i-1}{\mu-1}, \frac{1-(i-1)}{\mu-1}\right)$, $i \in [\![1, 50]\!]$. For Walsh surrogates, a Lasso regression is used to fit the model [25]. The maximum value allowed for the Walsh order is set to $d_{\max} = 3$ which corresponds to a cubic model. Recall that when analyzing the S-MCO variants with the $\mathsf{Order}_{\mathsf{static}}$ strategy, the actual order d used for training is a user-defined parameter, which we set in the range $d \in \{1, 2, 3 = d_{\max}\}$, i.e., three $d-$values are experimented for each variant using $\mathsf{Order}_{\mathsf{static}}$. The greedy strategy to adapt the Walsh order $\mathsf{Order}_{\mathsf{greedy}}$ uses a window of size $t = 5$. Each surrogate optimizer comes with different parameters, whose setting is given below. $\mathsf{Optim}_{\mathsf{MOEA/D}}$ is executed for 10 generations, as in [10]. It uses a one-point crossover followed by a uniform bit-flip mutation with a rate of $1/n$. The solution neighborhood considered in $\mathsf{Optim}_{\mathsf{MLS}}$ and $\mathsf{Optim}_{\mathsf{PLS}}$ is based on the bit-flip operator; i.e., two solutions are neighbors if their Hamming distance is 1. At last, $\mathsf{Optim}_{\mathsf{MLS}}$ uses the same 50 weight vectors as defined previously.

C. Performance Evaluation

Overall, we experiment 60 possible configurations for the S-MCO framework (See Table I for a summary): 3 surrogate optimizers $\times$ 4 selection strategies $\times$ $(3 + 2)$ Walsh order settings, together with 3 additional surrogate-less algorithms. Each algorithm is independently executed 10 times on the 8 considered MCOP instances (2×3 MNK-landscapes $+$ 2 MUBQP), for a maximum budget of $1\,500$ (expensive) evaluations and not exceeding 64 CPU hours per run. The experiments were conducted in Python 3.7 on an Intel Core Xeon E5-2630 (2.20 GHZ, 256 GB RAM) running under Debian 10. Notice that we shall report the approximation quality for different intermediate budgets, so as to render the relative anytime performance of all configurations.

For performance assessment, we use the additive epsilon indicator [31]. Notice, however, that our results were found to be consistent when using the hypervolume indicator, which we do not show in the reminder given our large set of data and for a better clarity of the presentation. The epsilon indicator gives the minimum factor by which an approximation set has to be translated in the objective space in order to (weakly) dominate a reference set. The reference set is constructed by merging the solutions found over all runs and configurations for a given instance, and removing dominated ones. For a given run, the archive of all non-dominated solutions found during the search process is used to measure approximation quality.

V. EXPERIMENTAL RESULTS AND ANALYSIS

In this section, we provide a detailed analysis of the proposed framework. Since S-MCO follows a plug-and-play scheme with 60 experimented configurations in this paper, a main challenge is to fairly address the impact of the combined effect of the designed components on search performance. In fact, as our empirical findings will reveal, there is a non-trivial interaction between the different components which we aim at describing in a comprehensive manner. In Section V-A,

we start by studying the interaction between the surrogate optimizer (Component #1) and the selection strategy (Component #2). In Section V-B, we address the relative performance of the different S-MCO variants when the best combination of these two components is considered. In Section V-C, the impact of the Walsh order setting (Component #3) is studied. In Section V-D, we finally report the relative performance of the best obtained S-MCO configuration against surrogate-less approaches.

A. Impact of Selection on Surrogate Optimizers

In the following, we address the relative impact of the different selection strategies (Component #2) on each *individual* S-MCO variant obtained with a particular surrogate optimizer (Component #1). This is summarized in Tables IV and III for MNK-landscapes and MUBQP, respectively. The tables show the relative rank of each selection strategy and each *static* setting of the Walsh order. Notice that different budgets, ranging from a small to a medium and a relatively high number of true evaluations, are considered. This is to better render the *anytime* behavior of S-MCO variants. It is important to remark that the ranks shown in Tables IV and III still do not allow to elicit in a comprehensive manner what is the best configuration of the S-MCO framework overall. However, we still show the average epsilon indicator value obtained for each variant (in parentheses), which is to provide the reader with a preliminary idea of the overall relative performance. A more specific analysis is to follow in the subsequent sections, after analyzing the relative impact of the selection strategy when combined with a particular surrogate optimizer. For now, and from Tables IV and III, we can draw two main general observations.

Firstly, the relative rank obtained by a given selection strategy is overall consistent over the different considered instances, although for MNK-landscapes, the value of k, which is the degree of non-linearity/ruggedness of the landscape, is found to impact the relative performance. This is actually related to the Walsh order setting. In fact, we can observe that the Walsh order d providing the best rank consistently depends on the value of k, such that $d = k+1$. For MUBQP instances, and since the underlying problem is quadratic, a Walsh order of 2 is always performing better. This is with no surprise since performance is tightly related to the ability of the Walsh surrogate to faithfully approximate the underlying landscape, which is directly related to the choice of the Walsh order. Overall, we can hence conclude that the rank obtained when using a surrogate optimizer with a particular selection strategy changes consistently with the choice of the order, suggesting that there is an optimal setting that depends on the underlying landscape. Interestingly, we can also observe that the ranks vary depending on the considered budget. This indicates that the different variants of the S-MCO framework might expose different anytime behaviors. This is to be analyzed in more details afterwards, in light of the other proposed *dynamic* strategies for setting the Walsh order.

Secondly, we can clearly see that the selection strategy providing the best rank is *different* depending on which surrogate optimizer is considered. In other words, for optimal performance, the selection strategy of the S-MCO framework has to be configured differently depending on the adopted surrogate optimizer. More specifically, with respect to the two local search optimizers ($\mathsf{Optim}_{\mathsf{PLS}}$ and $\mathsf{Optim}_{\mathsf{MLS}}$), we found that they perform at their best when using the $\mathsf{Select}_{\mathsf{BI}_{\mathsf{norm}}}$ strategy for MNK-landscapes with $k = 0$ and $k = 1$, whereas using the $\mathsf{Select}_{\mathsf{local}}$ strategy is a slightly better choice for these two optimizers when $k = 2$. For MUBQP, these two optimizers perform at their best when using different selection strategies: $\mathsf{Optim}_{\mathsf{PLS}}$ performs better with the $\mathsf{Select}_{\mathsf{local}}$ strategy, and $\mathsf{Optim}_{\mathsf{MLS}}$ performs better with the $\mathsf{Select}_{\mathsf{BI}_{\mathsf{norm}}}$ and $\mathsf{Select}_{\mathsf{BI}}$ strategies. With respect to the $\mathsf{Optim}_{\mathsf{MOEA/D}}$ optimizer, the $\mathsf{Select}_{\mathsf{BI}}$ strategy is the best performing one for MNK-landscapes, whereas the $\mathsf{Select}_{\mathsf{local}}$ strategy is better for MUBQP. We also found that these two selection strategies are definitely a better choice when using $\mathsf{Optim}_{\mathsf{MOEA/D}}$ compared to the $\mathsf{Select}_{\mathsf{BI}_{\mathsf{norm}}}$ and $\mathsf{Select}_{\mathsf{global}}$ strategies. Notice that these finding are to be contrasted with our preliminary work in [10], where $\mathsf{Optim}_{\mathsf{MOEA/D}}$ and the $\mathsf{Select}_{\mathsf{local}}$ strategies were recommended; i.e., we here provide evidence that other selection strategies are more accurate, depending on the considered problems.

To summarize these complex dependencies, we provide in Table II an overview of the relative performance of the selection strategies for the different surrogate optimizers, obtained by sorting the selection strategies according to the average ranks from Tables IV and III over the considered instances and budgets (as given by the 'avg. rank' rows). Looking at the impact of the selection strategy over the different surrogate optimizers, it is interesting to remark that for $\mathsf{Optim}_{\mathsf{PLS}}$ and $\mathsf{Optim}_{\mathsf{MLS}}$, the $\mathsf{Select}_{\mathsf{BI}_{\mathsf{norm}}}$ strategy is to be preferred, with an average rank of 1.40 and 1.29 respectively, whereas its average rank is of 4.36 when used in combination with $\mathsf{Optim}_{\mathsf{MOEA/D}}$. In other words, the $\mathsf{Select}_{\mathsf{BI}_{\mathsf{norm}}}$ strategy can be viewed as a relatively good selection strategy, except for $\mathsf{Optim}_{\mathsf{MOEA/D}}$. Besides, the $\mathsf{Select}_{\mathsf{global}}$ strategy shows the worst results overall and independently of the considered surrogate optimizer.

In the rest of the paper, and unless explicitly stated, we shall always consider the best performing selection strategy when dealing with a particular surrogate optimizer. We shall then address the relative performance of every S-MCO variant using a particular surrogate optimizer in combination with its best performing selection strategy, as depicted in Table II.

B. Impact of the Surrogate Optimizer

In Figures 1 and 2, we show the convergence (anytime) profile of the different S-MCO variants for MNK-landscapes and MUBQP, respectively. We here aim at analyzing the relative approximation quality obtained with the different surrogate optimizers, and the different static settings of the Walsh order.

As a first observation, we clearly see that the $\mathsf{Optim}_{\mathsf{MLS}}$ and $\mathsf{Optim}_{\mathsf{PLS}}$ optimizers have significantly better convergence profiles compared against $\mathsf{Optim}_{\mathsf{MOEA/D}}$. This is consistent over all considered instances, with very few exceptions. Interestingly, one shall recall here that the S-MCO framework

Table II
SORTING SELECTION STRATEGIES BY AVERAGE RANK (L = $\textsf{Select}_{\textsf{local}}$, G = $\textsf{Select}_{\textsf{global}}$, B_C = $\textsf{Select}_{\textsf{BI}}$, B_N = $\textsf{Select}_{\textsf{BI}_{\textsf{norm}}}$).

		$\textsf{Optim}_{\textsf{MOEA/D}}$	$\textsf{Optim}_{\textsf{MLS}}$	$\textsf{Optim}_{\textsf{PLS}}$
MNK-landscapes	$k=0$	$L \prec B_C \prec B_N \prec G$	$B_N \prec B_C \prec L \prec G$	$B_N \prec L \prec B_C \prec G$
MNK-landscapes	$k=1$	$B_C \prec L \prec B_N \prec G$	$B_N \prec B_C \prec L \prec G$	$B_N \prec L \prec B_C \prec G$
MNK-landscapes	$k=2$	$B_C \prec L \prec B_N \prec G$	$L \prec B_C \prec B_N \prec G$	$L \prec B_C \prec B_N \prec G$
MUBQP		$L \prec B_C \prec G \prec B_N$	$B_N \prec B_C \prec L \prec G$	$L \prec B_N \prec B_C \prec G$

Table III
RANK AND AVERAGE EPSILON INDICATOR VALUE ($\times 10^2$, BETWEEN BRACKETS) OF THE DIFFERENT ALGORITHMS AFTER 200, 700, AND 1 500 EVALUATIONS FOR MUBQP INSTANCES. FOR EACH BUDGET AND INSTANCE, A RANK c INDICATES THAT THE CORRESPONDING ALGORITHM WAS FOUND TO BE SIGNIFICANTLY OUTPERFORMED BY c OTHER STRATEGIES W.R.T. A WILCOXON STATISTICAL TEST WITH THE BONFERRONI CORRECTION AT A SIGNIFICANCE LEVEL OF 0.05. RANKS IN BOLD CORRESPOND TO APPROACHES THAT ARE NOT SIGNIFICANTLY OUTPERFORMED BY ANY OTHER FOR THE CONSIDERED BUDGET AND ORDER. UNDERLINED VALUES CORRESPONDS TO THE BEST APPROACH ON AVERAGE.

			$\textsf{Select}_{\textsf{local}}$			$\textsf{Select}_{\textsf{global}}$			$\textsf{Select}_{\textsf{BI}}$			$\textsf{Select}_{\textsf{BI}_{\textsf{norm}}}$		
			$\mathcal{O}$=1	$\mathcal{O}$=2	$\mathcal{O}$=3	$\mathcal{O}$=1	$\mathcal{O}$=2	$\mathcal{O}$=3	$\mathcal{O}$=1	$\mathcal{O}$=2	$\mathcal{O}$=3	$\mathcal{O}$=1	$\mathcal{O}$=2	$\mathcal{O}$=3
$\textsf{Optim}_{\textsf{MOEA/D}}$	$n=25$	200 evaluations	$1_{(371.5)}$	$\mathbf{0}_{(194.6)}$	$\mathbf{0}_{(252.2)}$	$5_{(756.2)}$	$4_{(639.8)}$	$3_{(658.9)}$	$3_{(515.5)}$	$\mathbf{0}_{(\underline{179.3})}$	$\mathbf{0}_{(234.9)}$	$5_{(776.2)}$	$4_{(666.2)}$	$5_{(709.6)}$
		700 evaluations	$4_{(280.7)}$	$\mathbf{0}_{(\underline{33.7})}$	$1_{(112)}$	$6_{(638.3)}$	$4_{(499.9)}$	$4_{(388.3)}$	$4_{(389.5)}$	$\mathbf{0}_{(55.8)}$	$1_{(98)}$	$5_{(663.9)}$	$4_{(565.1)}$	$4_{(476)}$
		1 500 evaluations	$4_{(260)}$	$\mathbf{0}_{(\underline{24.6})}$	$\mathbf{0}_{(49.7)}$	$8_{(576.6)}$	$4_{(344.4)}$	$4_{(285.6)}$	$4_{(370.6)}$	$1_{(47.6)}$	$\mathbf{0}_{(46.8)}$	$8_{(577.3)}$	$4_{(528.6)}$	$4_{(278.7)}$
	$n=50$	200 evaluations	$\mathbf{0}_{(\underline{1421.3})}$	$\mathbf{0}_{(1853.3)}$	$\mathbf{0}_{(1841.4)}$	$6_{(3302.9)}$	$6_{(2957.1)}$	$6_{(2986.2)}$	$\mathbf{0}_{(1551.5)}$	$\mathbf{0}_{(1784.6)}$	$\mathbf{0}_{(1728.7)}$	$6_{(3324.9)}$	$6_{(2860.8)}$	$6_{(3027.2)}$
		700 evaluations	$\mathbf{0}_{(\underline{935.6})}$	$\mathbf{0}_{(1063.3)}$	$\mathbf{0}_{(1031.7)}$	$6_{(2713.7)}$	$6_{(2359.3)}$	$5_{(2079.5)}$	$\mathbf{0}_{(937.2)}$	$\mathbf{0}_{(1050)}$	$\mathbf{0}_{(978.8)}$	$7_{(3003.5)}$	$6_{(2549.3)}$	$5_{(2277.9)}$
		1 500 evaluations	$1_{(826.4)}$	$\mathbf{0}_{(\underline{564.6})}$	$1_{(883.3)}$	$6_{(2222.2)}$	$6_{(1664.4)}$	$2_{(1403.7)}$	$1_{(829.2)}$	$\mathbf{0}_{(668.9)}$	$1_{(937.1)}$	$8_{(2762.7)}$	$7_{(2360.7)}$	$1_{(1637.8)}$
		avg. rank		**0.67**			5.06			0.83			5.28	
$\textsf{Optim}_{\textsf{MLS}}$	$n=25$	200 evaluations	$\mathbf{0}_{(290.1)}$	$\mathbf{0}_{(399.1)}$	$2_{(407.8)}$	$10_{(707.4)}$	$2_{(424)}$	$2_{(469.3)}$	$\mathbf{0}_{(360.2)}$	$\mathbf{0}_{(333.2)}$	$\mathbf{0}_{(335.2)}$	$\mathbf{0}_{(431.7)}$	$\mathbf{0}_{(\underline{233.6})}$	$\mathbf{0}_{(315.6)}$
		700 evaluations	$5_{(241.7)}$	$1_{(71.8)}$	$5_{(239.1)}$	$11_{(648.2)}$	$7_{(412.5)}$	$6_{(350.6)}$	$6_{(339.6)}$	$\mathbf{0}_{(48.2)}$	$1_{(108.1)}$	$5_{(404.7)}$	$\mathbf{0}_{(\underline{16.3})}$	$1_{(59.7)}$
		1 500 evaluations	$5_{(239.4)}$	$1_{(38.3)}$	$2_{(90.4)}$	$11_{(584.1)}$	$7_{(408.6)}$	$6_{(327.2)}$	$6_{(333.1)}$	$\mathbf{0}_{(36.3)}$	$1_{(57.3)}$	$6_{(383.3)}$	$\mathbf{0}_{(\underline{14.9})}$	$1_{(31.5)}$
	$n=50$	200 evaluations	$\mathbf{0}_{(\underline{1132.8})}$	$3_{(2800.1)}$	$3_{(2670.3)}$	$3_{(2352.1)}$	$3_{(2661.8)}$	$3_{(2775.1)}$	$\mathbf{0}_{(1260.6)}$	$3_{(3077.3)}$	$3_{(2741.5)}$	$\mathbf{0}_{(1239.1)}$	$3_{(2602.7)}$	$3_{(2823.9)}$
		700 evaluations	$\mathbf{0}_{(\underline{602.4})}$	$3_{(1562)}$	—	$7_{(2309.4)}$	$3_{(1585)}$	—	$1_{(857.1)}$	$3_{(1373)}$	—	$1_{(830.1)}$	$3_{(1598.5)}$	—
		1 500 evaluations	$2_{(557.9)}$	$2_{(793.6)}$	—	$7_{(2273.3)}$	$6_{(1401.1)}$	—	$3_{(856.7)}$	$\mathbf{0}_{(149.2)}$	—	$3_{(830.1)}$	$\mathbf{0}_{(\underline{115.1})}$	—
		avg. rank		1.89			5.22			1.5			**1.44**	
$\textsf{Optim}_{\textsf{PLS}}$	$n=25$	200 evaluations	$\mathbf{0}_{(343.9)}$	$\mathbf{0}_{(223)}$	$\mathbf{0}_{(248.4)}$	$11_{(765.4)}$	$4_{(516.3)}$	$1_{(459.7)}$	$1_{(409.6)}$	$\mathbf{0}_{(\underline{201.5})}$	$\mathbf{0}_{(269.4)}$	$2_{(440.3)}$	$\mathbf{0}_{(341.2)}$	$\mathbf{0}_{(363.6)}$
		700 evaluations	$6_{(272.3)}$	$\mathbf{0}_{(\underline{19.4})}$	$1_{(116.7)}$	$10_{(644.9)}$	$9_{(486.2)}$	$6_{(391.7)}$	$6_{(310.7)}$	$1_{(69.8)}$	$1_{(74.9)}$	$6_{(320.2)}$	$\mathbf{0}_{(55.4)}$	$1_{(107.1)}$
		1 500 evaluations	$6_{(264.3)}$	$\mathbf{0}_{(\underline{15.3})}$	$\mathbf{0}_{(50.4)}$	$11_{(606)}$	$9_{(470)}$	$6_{(369.6)}$	$6_{(287.5)}$	$1_{(69.8)}$	$1_{(58.5)}$	$6_{(307.9)}$	$\mathbf{0}_{(55.4)}$	$1_{(68.4)}$
	$n=50$	200 evaluations	$\mathbf{0}_{(\underline{1187.5})}$	$3_{(2139.8)}$	$3_{(2106.1)}$	$8_{(2939)}$	$3_{(2471.9)}$	$3_{(2410.4)}$	$\mathbf{0}_{(1308.5)}$	$3_{(2352.7)}$	$3_{(2403.8)}$	$\mathbf{0}_{(1309.1)}$	$3_{(2471.9)}$	$3_{(2410.4)}$
		700 evaluations	$\mathbf{0}_{(\underline{781.3})}$	$\mathbf{0}_{(1049.5)}$	—	$7_{(2726)}$	$6_{(1588.1)}$	—	$1_{(1118.7)}$	$\mathbf{0}_{(899.5)}$	—	$\mathbf{0}_{(1015.3)}$	$\mathbf{0}_{(945.9)}$	—
		1 500 evaluations	$3_{(734.3)}$	$\mathbf{0}_{(142.1)}$	—	$7_{(2600.6)}$	$6_{(1588)}$	—	$3_{(1054.8)}$	$\mathbf{0}_{(177.8)}$	—	$3_{(960.5)}$	$\mathbf{0}_{(\underline{115.3})}$	—
		avg. rank		**1.22**			5.94			1.5			1.39	

uses decomposition to both structure the main population of evaluated solutions, and to select the next solution to evaluate, independently of the inner surrogate optimizer. Hence, this first observation shows that (i) the S-MCO framework does *not* necessarily need to be configured with a decomposition-based inner optimizer to handle the underlying surrogate, but more importantly that (ii) other more accurate optimizers should be used specifically to the underlying Walsh surrogate. This is again to be contrasted with our preliminary work [10], where MOEA/D was employed at the different design stages. To push our discussion further, we found that the only reason that may motivate the use of MOEA/D as an inner optimizer is its extremely efficient computational complexity. In fact, as can be seen in Figure 2, when dealing with the largest MUBQP instance with $n = 50$, and using the most computational demanding Walsh order $d = 3$, the S-MCO framework is able to complete all iterations (up to 1 500 true evaluations) within the maximum allowed CPU time, *only* when configured with the $\textsf{Optim}_{\textsf{MOEA/D}}$ optimizer. This indicates that $\textsf{Optim}_{\textsf{MOEA/D}}$ may be a reasonable choice only if there exists a CPU time trade-off between the function evaluation cost on one hand, and the optimization process on the other hand. In other words, unless the (true) evaluation function is *not* too expensive, the $\textsf{Optim}_{\textsf{MLS}}$ and $\textsf{Optim}_{\textsf{PLS}}$ optimizers should be preferred over the $\textsf{Optim}_{\textsf{MOEA/D}}$ optimizer. Notice that studying in more details such a trade-off is out of the scope of this paper, but is worth to be investigated in future studies.

Let us now analyze in more details the two $\textsf{Optim}_{\textsf{MLS}}$ and $\textsf{Optim}_{\textsf{PLS}}$ optimizers, which were found to lead to a better approximation quality. Although overall they expose a similar behavior, their relative anytime profile depends both on the difficulty of the tackled problem, and more importantly on the complexity of the Walsh surrogate, as implied by the choice of the Walsh order. This is discussed in the next paragraphs by splitting our experimented instances into three classes: (i) linear, i.e., MNK-landscapes with $k = 0$, (ii) quadratic, i.e., MNK-landscapes with $k = 1$ and MUBQP, and (iii) cubic, i.e., MNK-landscapes with $k = 2$. Notice that the order of the (unknown) exact Walsh transform of any function is at most 1, 2, and 3, respectively for these classes.

For the linear instances, $\textsf{Optim}_{\textsf{PLS}}$ used in combination with the (exact) Walsh order of 1, has a better anytime behavior than $\textsf{Optim}_{\textsf{MLS}}$, i.e., $\textsf{Optim}_{\textsf{PLS}}$ is converging very quickly to a high-quality approximation set, and $\textsf{Optim}_{\textsf{MLS}}$ is only able to obtain a better quality when a higher number of evaluations is affordable. When using an overestimated Walsh order of 2

Table IV

RANK AND AVERAGE EPSILON INDICATOR VALUE ($\times 10^2$, BETWEEN BRACKETS) OF THE DIFFERENT ALGORITHMS AFTER 200, 700, AND 1 500 EVALUATIONS FOR MNK-LANDSCAPES. FOR EACH BUDGET AND INSTANCE, A RANK c INDICATES THAT THE CORRESPONDING ALGORITHM WAS FOUND TO BE SIGNIFICANTLY OUTPERFORMED BY c OTHER ALGORITHMS IN THE CORRESPONDING ROW, AND W.R.T. A WILCOXON STATISTICAL TEST WITH THE BONFERRONI CORRECTION AT A SIGNIFICANCE LEVEL OF 0.05. RANKS IN BOLD CORRESPOND TO APPROACHES THAT ARE NOT SIGNIFICANTLY OUTPERFORMED BY ANY OTHER FOR THE CONSIDERED BUDGET AND ORDER. UNDERLINED VALUES CORRESPONDS TO THE BEST APPROACH ON AVERAGE.

				$\text{Select}_{\text{local}}$			$\text{Select}_{\text{global}}$			$\text{Select}_{\text{BI}}$			$\text{Select}_{\text{BI}_{\text{norm}}}$		
				$\mathcal{O}$=1	$\mathcal{O}$=2	$\mathcal{O}$=3	$\mathcal{O}$=1	$\mathcal{O}$=2	$\mathcal{O}$=3	$\mathcal{O}$=1	$\mathcal{O}$=2	$\mathcal{O}$=3	$\mathcal{O}$=1	$\mathcal{O}$=2	$\mathcal{O}$=3
$\text{Optim}_{\text{MOEA/D}}$	$k=0$	$n=25$	200 evaluations	$\mathbf{0}_{(1.4)}$	$\mathbf{0}_{(1.8)}$	$3_{(2.4)}$	$9_{(5)}$	$9_{(5.4)}$	$9_{(5.7)}$	$\mathbf{0}_{(1)}$	$\mathbf{0}_{(1.2)}$	$\mathbf{0}_{(1.8)}$	$\mathbf{0}_{(1.3)}$	$\mathbf{0}_{(1.4)}$	$3_{(2.4)}$
			700 evaluations	$\mathbf{0}_{(1)}$	$\mathbf{0}_{(1.1)}$	$2_{(1.3)}$	$9_{(4.4)}$	$9_{(4.1)}$	$9_{(4.1)}$	$\mathbf{0}_{(0.7)}$	$\mathbf{0}_{(0.7)}$	$\mathbf{0}_{(0.9)}$	$\mathbf{0}_{(1.1)}$	$\mathbf{0}_{(1)}$	$\mathbf{0}_{(1.1)}$
			1 500 evaluations	$\mathbf{0}_{(0.8)}$	$\mathbf{0}_{(0.8)}$	$\mathbf{0}_{(0.9)}$	$9_{(3.9)}$	$9_{(3.6)}$	$9_{(3.2)}$	$\mathbf{0}_{(0.6)}$	$\mathbf{0}_{(0.6)}$	$\mathbf{0}_{(0.7)}$	$\mathbf{0}_{(0.8)}$	$\mathbf{0}_{(0.9)}$	$\mathbf{0}_{(0.9)}$
		$n=50$	200 evaluations	$\mathbf{0}_{(2)}$	$2_{(3.7)}$	$2_{(5)}$	$2_{(7.9)}$	$6_{(8.6)}$	$6_{(9.2)}$	$\mathbf{0}_{(2.3)}$	$1_{(3.6)}$	$2_{(4.7)}$	$6_{(8.6)}$	$6_{(9.1)}$	$6_{(9.1)}$
			700 evaluations	$\mathbf{0}_{(1.4)}$	$\mathbf{0}_{(1.8)}$	$1_{(2.3)}$	$5_{(6.5)}$	$6_{(6.3)}$	$6_{(8)}$	$\mathbf{0}_{(2)}$	$1_{(2.3)}$	$4_{(3)}$	$6_{(8.3)}$	$6_{(8.4)}$	$6_{(8.5)}$
			1 500 evaluations	$\mathbf{0}_{(1.1)}$	$\mathbf{0}_{(1.3)}$	$\mathbf{0}_{(1.4)}$	$6_{(3.9)}$	$5_{(4.2)}$	$6_{(4.9)}$	$\mathbf{0}_{(1.7)}$	$1_{(1.9)}$	$3_{(2.4)}$	$8_{(7.6)}$	$8_{(7.6)}$	$8_{(7.7)}$
			avg. rank		**0.56**			7.17			0.67			3.5	
	$k=1$	$n=25$	200 evaluations	$3_{(4.2)}$	$\mathbf{0}_{(1.5)}$	$\mathbf{0}_{(2.6)}$	$8_{(8.7)}$	$6_{(6.8)}$	$8_{(7.7)}$	$3_{(4.5)}$	$\mathbf{0}_{(1.1)}$	$\mathbf{0}_{(2.5)}$	$3_{(4.2)}$	$8_{(7.2)}$	$\mathbf{0}_{(1.7)}$
			700 evaluations	$5_{(3.4)}$	$\mathbf{0}_{(0.5)}$	$\mathbf{0}_{(0.5)}$	$7_{(6.6)}$	$6_{(5.7)}$	$7_{(6.2)}$	$5_{(3.8)}$	$\mathbf{0}_{(0.5)}$	$\mathbf{0}_{(0.6)}$	$5_{(2.8)}$	$7_{(6)}$	$\mathbf{0}_{(0.6)}$
			1 500 evaluations	$5_{(2.9)}$	$\mathbf{0}_{(0.3)}$	$\mathbf{0}_{(0.4)}$	$7_{(5.7)}$	$5_{(4.4)}$	$6_{(4.8)}$	$5_{(3.5)}$	$\mathbf{0}_{(0.4)}$	$\mathbf{0}_{(0.4)}$	$5_{(2.6)}$	$7_{(5.7)}$	$\mathbf{0}_{(0.4)}$
		$n=50$	200 evaluations	$\mathbf{0}_{(5.2)}$	$\mathbf{0}_{(4.8)}$	$1_{(6.3)}$	$6_{(9.7)}$	$6_{(10.6)}$	$6_{(10.9)}$	$\mathbf{0}_{(5.8)}$	$\mathbf{0}_{(4.8)}$	$\mathbf{0}_{(5.6)}$	$6_{(9.8)}$	$6_{(10.7)}$	$6_{(10.9)}$
			700 evaluations	$1_{(4.1)}$	$\mathbf{0}_{(3.2)}$	$\mathbf{0}_{(3.7)}$	$6_{(7.7)}$	$5_{(7.7)}$	$6_{(9.1)}$	$1_{(4.4)}$	$\mathbf{0}_{(2.8)}$	$\mathbf{0}_{(3.4)}$	$6_{(8.7)}$	$6_{(9.6)}$	$7_{(9.9)}$
			1 500 evaluations	$1_{(3.8)}$	$\mathbf{0}_{(2.9)}$	$\mathbf{0}_{(3.1)}$	$4_{(6.5)}$	$\mathbf{0}_{(5.6)}$	$3_{(6)}$	$\mathbf{0}_{(3.9)}$	$\mathbf{0}_{(2.8)}$	$\mathbf{0}_{(2.8)}$	$6_{(8.3)}$	$6_{(8.6)}$	$6_{(8.7)}$
			avg. rank		0.89			5.67			**0.78**			5	
	$k=2$	$n=25$	200 evaluations	$4_{(8)}$	$\mathbf{0}_{(5.6)}$	$\mathbf{0}_{(5.9)}$	$6_{(9)}$	$6_{(8.3)}$	$4_{(8.4)}$	$2_{(7.5)}$	$\mathbf{0}_{(4.3)}$	$\mathbf{0}_{(4.7)}$	$5_{(8.2)}$	$\mathbf{0}_{(4.7)}$	$\mathbf{0}_{(5.2)}$
			700 evaluations	$6_{(6.5)}$	$\mathbf{0}_{(3.9)}$	$\mathbf{0}_{(3)}$	$6_{(7.7)}$	$1_{(5.4)}$	$\mathbf{0}_{(6.2)}$	$6_{(7.2)}$	$\mathbf{0}_{(2.9)}$	$\mathbf{0}_{(2.7)}$	$5_{(6.8)}$	$\mathbf{0}_{(3.6)}$	$\mathbf{0}_{(3.2)}$
			1 500 evaluations	$6_{(6.2)}$	$\mathbf{0}_{(3.4)}$	$\mathbf{0}_{(2)}$	$6_{(7.2)}$	$\mathbf{0}_{(5)}$	$\mathbf{0}_{(5.3)}$	$5_{(5.9)}$	$\mathbf{0}_{(2.4)}$	$\mathbf{0}_{(2.1)}$	$5_{(6.1)}$	$\mathbf{0}_{(2.8)}$	$\mathbf{0}_{(2.7)}$
		$n=50$	200 evaluations	$\mathbf{0}_{(8)}$	$\mathbf{0}_{(7.7)}$	$\mathbf{0}_{(7.7)}$	$5_{(13.3)}$	$5_{(11.6)}$	$5_{(12.3)}$	$\mathbf{0}_{(9.4)}$	$\mathbf{0}_{(6.5)}$	$\mathbf{0}_{(7.3)}$	$5_{(13.3)}$	$5_{(11.6)}$	$5_{(12.3)}$
			700 evaluations	$\mathbf{0}_{(5.7)}$	$\mathbf{0}_{(4.4)}$	$\mathbf{0}_{(4.6)}$	$6_{(11.2)}$	$5_{(9.6)}$	$6_{(10.6)}$	$\mathbf{0}_{(6.3)}$	$\mathbf{0}_{(4.6)}$	$\mathbf{0}_{(4.4)}$	$6_{(11.4)}$	$6_{(10.6)}$	$6_{(11.8)}$
			1 500 evaluations	$1_{(5.4)}$	$\mathbf{0}_{(3.4)}$	$\mathbf{0}_{(3.3)}$	$6_{(10.7)}$	$4_{(7.8)}$	$4_{(8.3)}$	$\mathbf{0}_{(5.3)}$	$\mathbf{0}_{(4.1)}$	$\mathbf{0}_{(3.7)}$	$6_{(10.9)}$	$6_{(9.6)}$	$6_{(10.5)}$
			avg. rank		0.94			4.17			**0.72**			3.67	
$\text{Optim}_{\text{MLS}}$	$k=0$	$n=25$	200 evaluations	$5_{(4.7)}$	$3_{(3.3)}$	$2_{(2.4)}$	$3_{(2.2)}$	$4_{(4.7)}$	$7_{(5.2)}$	$5_{(3.7)}$	$3_{(3.2)}$	$3_{(2.8)}$	$\mathbf{0}_{(1.1)}$	$\mathbf{0}_{(0.4)}$	$1_{(1)}$
			700 evaluations	$4_{(3.1)}$	$3_{(2.4)}$	$3_{(1.5)}$	$3_{(2.1)}$	$8_{(4.1)}$	$8_{(4.7)}$	$3_{(2.9)}$	$3_{(2.3)}$	$2_{(1.8)}$	$\mathbf{0}_{(0.3)}$	$\mathbf{0}_{(0.3)}$	$\mathbf{0}_{(0.3)}$
			1 500 evaluations	$3_{(2.3)}$	$3_{(1.7)}$	$\mathbf{0}_{(1.1)}$	$3_{(2.1)}$	$10_{(3.9)}$	$10_{(4)}$	$3_{(2.2)}$	$3_{(1.7)}$	$2_{(1.3)}$	$\mathbf{0}_{(0.3)}$	$\mathbf{0}_{(0.3)}$	$\mathbf{0}_{(0.3)}$
		$n=50$	200 evaluations	$3_{(5.6)}$	$1_{(4.6)}$	$1_{(4.9)}$	$1_{(4.1)}$	$5_{(6.8)}$	$10_{(7.4)}$	$1_{(3.7)}$	$1_{(4.1)}$	$2_{(5.4)}$	$1_{(3.9)}$	$\mathbf{0}_{(1.7)}$	$1_{(4.5)}$
			700 evaluations	$9_{(4.3)}$	$1_{(1.2)}$	$3_{(1.2)}$	$4_{(1.6)}$	$10_{(6.2)}$	$10_{(6.4)}$	$5_{(1.9)}$	$\mathbf{0}_{(0.6)}$	$2_{(1)}$	$\mathbf{0}_{(0.4)}$	$\mathbf{0}_{(0.3)}$	$\mathbf{0}_{(0.5)}$
			1 500 evaluations	$9_{(3.3)}$	$1_{(0.5)}$	$\mathbf{0}_{(0.3)}$	$8_{(1.6)}$	$10_{(5.9)}$	$10_{(6.2)}$	$6_{(0.8)}$	$\mathbf{0}_{(0.3)}$	$\mathbf{0}_{(0.2)}$	$\mathbf{0}_{(0.3)}$	$\mathbf{0}_{(0.3)}$	$\mathbf{0}_{(0.3)}$
			avg. rank		3			6.89			2.44			**0.17**	
	$k=1$	$n=25$	200 evaluations	$2_{(3.9)}$	$1_{(2.7)}$	$1_{(2.8)}$	$6_{(6.3)}$	$4_{(5.4)}$	$6_{(6.3)}$	$3_{(4.8)}$	$1_{(1.5)}$	$1_{(3.2)}$	$4_{(4.7)}$	$\mathbf{0}_{(0.6)}$	$1_{(1.7)}$
			700 evaluations	$6_{(2.9)}$	$3_{(1.2)}$	$1_{(0.6)}$	$9_{(5.5)}$	$7_{(4.8)}$	$8_{(5.2)}$	$6_{(3.9)}$	$2_{(1)}$	$1_{(0.9)}$	$6_{(3.8)}$	$\mathbf{0}_{(0.4)}$	$\mathbf{0}_{(0.3)}$
			1 500 evaluations	$6_{(2.6)}$	$3_{(0.9)}$	$1_{(0.5)}$	$9_{(5.1)}$	$7_{(4.4)}$	$9_{(4.9)}$	$6_{(3.7)}$	$2_{(0.8)}$	$1_{(0.8)}$	$6_{(3.5)}$	$\mathbf{0}_{(0.4)}$	$\mathbf{0}_{(0.3)}$
		$n=50$	200 evaluations	$1_{(4.7)}$	$2_{(5.9)}$	$1_{(6.2)}$	$1_{(5.2)}$	$5_{(6.7)}$	$8_{(8.2)}$	$\mathbf{0}_{(4.3)}$	$1_{(5.3)}$	$3_{(6.1)}$	$\mathbf{0}_{(4.4)}$	$\mathbf{0}_{(3.2)}$	$\mathbf{0}_{(4.5)}$
			700 evaluations	$5_{(3.3)}$	$1_{(1.6)}$	$1_{(2.4)}$	$8_{(5.1)}$	$9_{(6.5)}$	$9_{(6.6)}$	$5_{(3.2)}$	$\mathbf{0}_{(0.8)}$	$2_{(1.4)}$	$5_{(3.5)}$	$\mathbf{0}_{(0.4)}$	$1_{(1.3)}$
			1 500 evaluations	$6_{(2.9)}$	$4_{(1)}$	$\mathbf{0}_{(0.6)}$	$9_{(5)}$	$9_{(6.4)}$	$9_{(6.5)}$	$6_{(3.2)}$	$\mathbf{0}_{(0.4)}$	$\mathbf{0}_{(0.4)}$	$6_{(3.4)}$	$\mathbf{0}_{(0.3)}$	$\mathbf{0}_{(0.4)}$
			avg. rank		2.5			7.33			2.22			**1.61**	
	$k=2$	$n=25$	200 evaluations	$\mathbf{0}_{(7.4)}$	$\mathbf{0}_{(6.4)}$	$\mathbf{0}_{(5.9)}$	$4_{(8.7)}$	$\mathbf{0}_{(5)}$	$1_{(7.1)}$	$1_{(8.5)}$	$\mathbf{0}_{(5.7)}$	$\mathbf{0}_{(6.1)}$	$1_{(8.3)}$	$\mathbf{0}_{(5.3)}$	$\mathbf{0}_{(5.5)}$
			700 evaluations	$4_{(5.6)}$	$3_{(2.8)}$	$\mathbf{0}_{(1.1)}$	$8_{(7.4)}$	$3_{(3.6)}$	$4_{(4.8)}$	$7_{(7.1)}$	$2_{(2.9)}$	$\mathbf{0}_{(0.9)}$	$7_{(7.3)}$	$2_{(2.5)}$	$\mathbf{0}_{(0.5)}$
			1 500 evaluations	$6_{(5.1)}$	$3_{(2)}$	$\mathbf{0}_{(0.7)}$	$8_{(7.1)}$	$6_{(3.5)}$	$6_{(4.6)}$	$7_{(6.6)}$	$3_{(2.1)}$	$\mathbf{0}_{(0.5)}$	$8_{(6.9)}$	$3_{(2)}$	$\mathbf{0}_{(0.5)}$
		$n=50$	200 evaluations	$\mathbf{0}_{(7)}$	$\mathbf{0}_{(7.7)}$	$\mathbf{0}_{(8.4)}$	$8_{(10.7)}$	$\mathbf{0}_{(8.8)}$	$1_{(9.5)}$	$\mathbf{0}_{(8.3)}$	$\mathbf{0}_{(7.3)}$	$\mathbf{0}_{(7.6)}$	$1_{(9.1)}$	$\mathbf{0}_{(7)}$	$\mathbf{0}_{(6.3)}$
			700 evaluations	$2_{(4.7)}$	$\mathbf{0}_{(3.6)}$	$\mathbf{0}_{(3.7)}$	$9_{(10)}$	$5_{(7.2)}$	$7_{(8.3)}$	$6_{(6.3)}$	$\mathbf{0}_{(3.2)}$	$\mathbf{0}_{(4)}$	$6_{(6.6)}$	$\mathbf{0}_{(2.8)}$	$\mathbf{0}_{(3.3)}$
			1 500 evaluations	$6_{(4.3)}$	$\mathbf{0}_{(1.9)}$	$\mathbf{0}_{(1.9)}$	$11_{(9.8)}$	$6_{(6.7)}$	$6_{(7.3)}$	$7_{(6.1)}$	$\mathbf{0}_{(1.8)}$	$\mathbf{0}_{(2)}$	$7_{(6)}$	$\mathbf{0}_{(1.4)}$	$\mathbf{0}_{(1.5)}$
			avg. rank		**1.33**			5.17			1.83			1.94	
$\text{Optim}_{\text{PLS}}$	$k=0$	$n=25$	200 evaluations	$\mathbf{0}_{(0.4)}$	$\mathbf{0}_{(0.4)}$	$6_{(1.7)}$	$10_{(6.4)}$	$9_{(5.4)}$	$9_{(4.9)}$	$\mathbf{0}_{(0.5)}$	$3_{(0.8)}$	$3_{(1.2)}$	$\mathbf{0}_{(0.4)}$	$\mathbf{0}_{(0.5)}$	$2_{(1)}$
			700 evaluations	$\mathbf{0}_{(0.4)}$	$\mathbf{0}_{(0.4)}$	$\mathbf{0}_{(0.3)}$	$9_{(5.4)}$	$9_{(4.6)}$	$9_{(4.4)}$	$1_{(0.5)}$	$4_{(0.8)}$	$4_{(1.1)}$	$\mathbf{0}_{(0.4)}$	$\mathbf{0}_{(0.5)}$	$\mathbf{0}_{(0.5)}$
			1 500 evaluations	$\mathbf{0}_{(0.4)}$	$\mathbf{0}_{(0.4)}$	$\mathbf{0}_{(0.3)}$	$10_{(4.7)}$	$9_{(4.3)}$	$9_{(3.9)}$	$1_{(0.5)}$	$4_{(0.8)}$	$4_{(1.1)}$	$\mathbf{0}_{(0.4)}$	$\mathbf{0}_{(0.5)}$	$\mathbf{0}_{(0.5)}$
		$n=50$	200 evaluations	$\mathbf{0}_{(0.4)}$	$3_{(2.3)}$	$6_{(4.7)}$	$9_{(8.1)}$	$9_{(6.9)}$	$7_{(6.6)}$	$\mathbf{0}_{(0.5)}$	$3_{(2.2)}$	$6_{(4.7)}$	$\mathbf{0}_{(0.4)}$	$\mathbf{0}_{(1.5)}$	$4_{(4.1)}$
			700 evaluations	$\mathbf{0}_{(0.4)}$	$\mathbf{0}_{(0.3)}$	$2_{(0.7)}$	$10_{(7.2)}$	$9_{(6.3)}$	$9_{(6)}$	$1_{(0.5)}$	$\mathbf{0}_{(0.6)}$	$4_{(0.8)}$	$\mathbf{0}_{(0.4)}$	$\mathbf{0}_{(0.3)}$	$1_{(0.5)}$
			1 500 evaluations	$\mathbf{0}_{(0.4)}$	$\mathbf{0}_{(0.3)}$	$\mathbf{0}_{(0.2)}$	$9_{(6.7)}$	$9_{(6.1)}$	$9_{(5.9)}$	$2_{(0.5)}$	$\mathbf{0}_{(0.6)}$	$6_{(0.7)}$	$\mathbf{0}_{(0.4)}$	$\mathbf{0}_{(0.3)}$	$\mathbf{0}_{(0.4)}$
			avg. rank		0.94			9.06			2.56			**0.39**	
	$k=1$	$n=25$	200 evaluations	$5_{(3.7)}$	$\mathbf{0}_{(0.8)}$	$1_{(1.5)}$	$8_{(6.8)}$	$8_{(6.5)}$	$7_{(6.1)}$	$6_{(5)}$	$\mathbf{0}_{(0.9)}$	$1_{(2.1)}$	$5_{(4.4)}$	$\mathbf{0}_{(0.5)}$	$1_{(1.3)}$
			700 evaluations	$6_{(2.8)}$	$\mathbf{0}_{(0.3)}$	$\mathbf{0}_{(0.3)}$	$8_{(5.6)}$	$8_{(5.6)}$	$7_{(5.5)}$	$7_{(4.6)}$	$\mathbf{0}_{(0.6)}$	$2_{(0.6)}$	$6_{(4.1)}$	$\mathbf{0}_{(0.4)}$	$\mathbf{0}_{(0.4)}$
			1 500 evaluations	$6_{(2.8)}$	$\mathbf{0}_{(0.3)}$	$\mathbf{0}_{(0.3)}$	$8_{(5.2)}$	$7_{(4.8)}$	$7_{(5)}$	$7_{(4.3)}$	$\mathbf{0}_{(0.6)}$	$3_{(0.6)}$	$6_{(4)}$	$\mathbf{0}_{(0.4)}$	$\mathbf{0}_{(0.4)}$
		$n=50$	200 evaluations	$1_{(4.6)}$	$\mathbf{0}_{(3.5)}$	$2_{(5.4)}$	$7_{(7.1)}$	$8_{(7.5)}$	$7_{(7.7)}$	$1_{(4.9)}$	$1_{(4)}$	$1_{(5.2)}$	$1_{(4.6)}$	$\mathbf{0}_{(2.4)}$	$1_{(4.3)}$
			700 evaluations	$6_{(3.5)}$	$\mathbf{0}_{(0.6)}$	$3_{(1.7)}$	$9_{(6.7)}$	$9_{(7.3)}$	$9_{(6.7)}$	$6_{(4.1)}$	$\mathbf{0}_{(0.6)}$	$3_{(1.9)}$	$6_{(4)}$	$\mathbf{0}_{(0.4)}$	$1_{(1)}$
			1 500 evaluations	$6_{(3.4)}$	$\mathbf{0}_{(0.4)}$	$\mathbf{0}_{(0.4)}$	$9_{(6.6)}$	$9_{(7.1)}$	$9_{(6.4)}$	$6_{(4)}$	$\mathbf{0}_{(0.6)}$	$\mathbf{0}_{(0.5)}$	$6_{(4)}$	$\mathbf{0}_{(0.4)}$	$\mathbf{0}_{(0.4)}$
			avg. rank		2			8			2.44			**1.83**	
	$k=2$	$n=25$	200 evaluations	$3_{(6.4)}$	$\mathbf{0}_{(4.8)}$	$\mathbf{0}_{(5)}$	$9_{(8.9)}$	$\mathbf{0}_{(5.8)}$	$\mathbf{0}_{(6.4)}$	$3_{(7.7)}$	$\mathbf{0}_{(3.9)}$	$\mathbf{0}_{(5.3)}$	$1_{(6.8)}$	$\mathbf{0}_{(3.8)}$	$\mathbf{0}_{(4)}$
			700 evaluations	$7_{(6)}$	$3_{(2.3)}$	$\mathbf{0}_{(1)}$	$8_{(8)}$	$4_{(3.6)}$	$4_{(4.7)}$	$5_{(5.9)}$	$3_{(2.5)}$	$\mathbf{0}_{(0.7)}$	$6_{(5.9)}$	$3_{(2.1)}$	$\mathbf{0}_{(0.7)}$
			1 500 evaluations	$6_{(5.7)}$	$3_{(1.8)}$	$\mathbf{0}_{(0.2)}$	$8_{(7.1)}$	$6_{(3.6)}$	$6_{(4.5)}$	$6_{(5.4)}$	$3_{(2.3)}$	$\mathbf{0}_{(0.6)}$	$7_{(5.7)}$	$3_{(1.9)}$	$\mathbf{0}_{(0.5)}$
		$n=50$	200 evaluations	$2_{(8)}$	$\mathbf{0}_{(5.6)}$	$\mathbf{0}_{(6.6)}$	$7_{(10.7)}$	$\mathbf{0}_{(8.1)}$	$5_{(9.5)}$	$5_{(9)}$	$\mathbf{0}_{(6.4)}$	$\mathbf{0}_{(6.7)}$	$4_{(8.5)}$	$\mathbf{0}_{(6.2)}$	$\mathbf{0}_{(5.9)}$
			700 evaluations	$4_{(5.9)}$	$\mathbf{0}_{(3)}$	$\mathbf{0}_{(3.8)}$	$10_{(9.9)}$	$4_{(6.6)}$	$6_{(7.8)}$	$4_{(6.2)}$	$\mathbf{0}_{(2.7)}$	$\mathbf{0}_{(4)}$	$6_{(6.3)}$	$\mathbf{0}_{(2.4)}$	$\mathbf{0}_{(3.3)}$
			1 500 evaluations	$6_{(5.2)}$	$\mathbf{0}_{(2.2)}$	$\mathbf{0}_{(2.3)}$	$11_{(9.3)}$	$6_{(5.6)}$	$7_{(7.2)}$	$6_{(5.7)}$	$\mathbf{0}_{(1.9)}$	$\mathbf{0}_{(2.6)}$	$6_{(6)}$	$\mathbf{0}_{(1.8)}$	$\mathbf{0}_{(1.8)}$
			avg. rank		**1.89**			5.61			1.94			2	

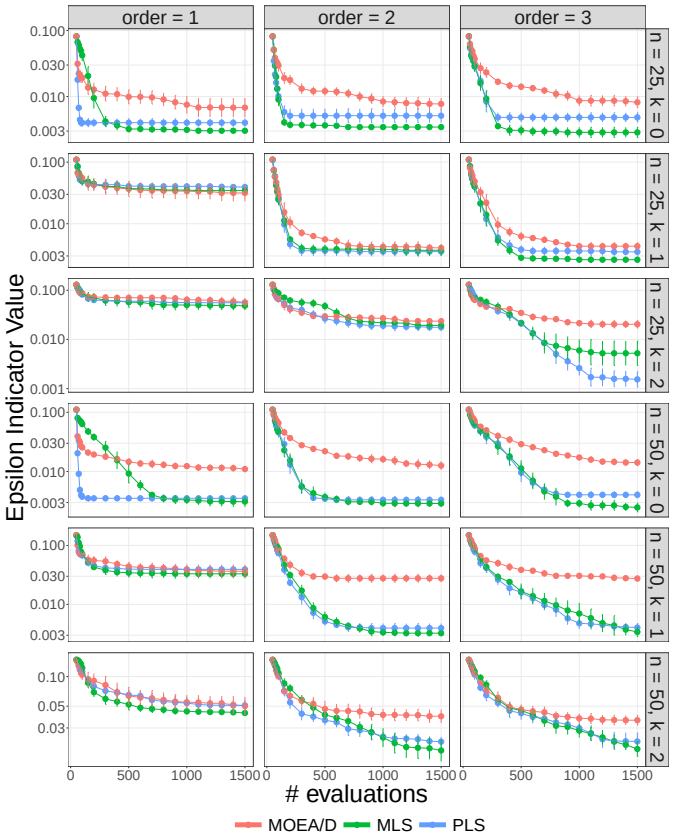

Figure 1. Convergence profile of S-MCO when using different surrogate optimizers with their corresponding best selection strategy on MNK-landscapes.

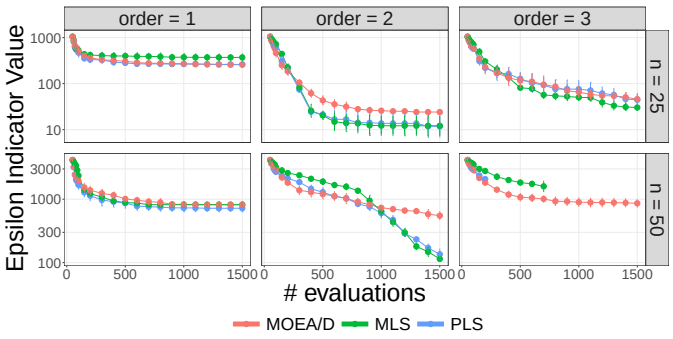

Figure 2. Convergence profile of S-MCO when using different surrogate optimizers with their corresponding best selection strategy on MUBQP instances.

or 3, the situation is clearly reversed, and $\text{Optim}_{\text{MLS}}$ is found to perform significantly better than $\text{Optim}_{\text{PLS}}$ independently of the available budget.

For quadratic instances, when setting the Walsh order to the exact transform order of 2, we found that $\text{Optim}_{\text{MLS}}$ and $\text{Optim}_{\text{PLS}}$ have seemingly the same convergence profile for $n = 25$. By contrast, for $n = 50$, $\text{Optim}_{\text{PLS}}$ is slightly better under a restricted budget, while $\text{Optim}_{\text{MLS}}$ is slightly better under higher budgets, which is to recall their behavior for linear instances. When the Walsh order is set to an underestimated value of 1, the approximation quality obtained with all optimizers drops down in comparison to an (exact) order of 2. Interestingly, using an underestimated order value

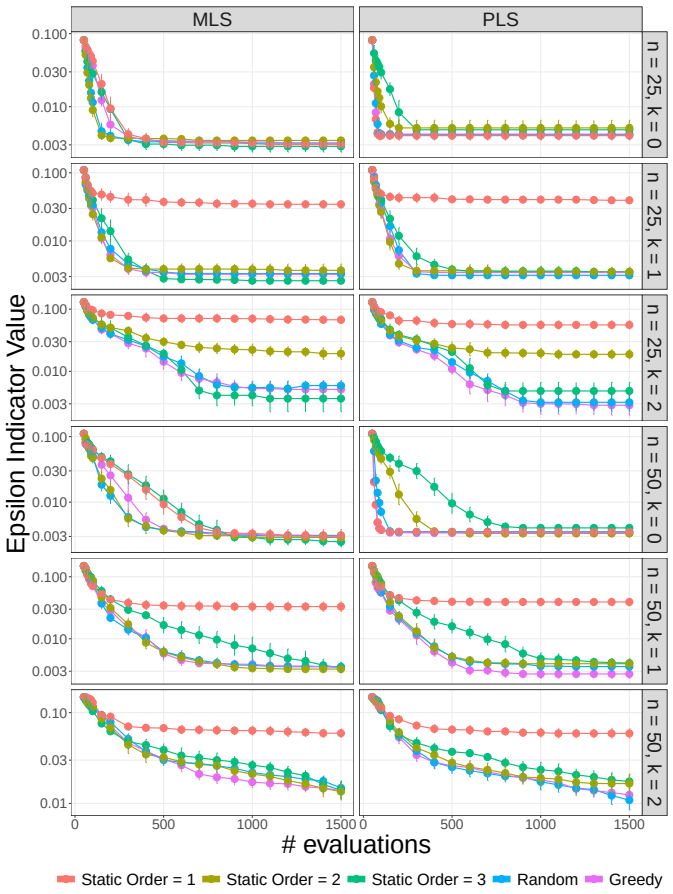

Figure 3. Convergence profile of S-MCO when using different Walsh order settings with the $\text{Select}_{\text{BI}_{\text{norm}}}$ strategy on MNK-landscapes.

does not prevent the S-MCO framework to keep improving during the very first iterations, that is, with a very restricted budget. When the Walsh order is set to an overestimated value of 3, the overall approximation quality of both optimizers seems to be comparable to the exact setting of the order.

Finally, for most complex cubic instances, we found that for $n = 25$, an underestimated order setting of 1 or 2 is not competitive with the exact setting of 3, independently of the surrogate optimizer. For problem dimension $n = 50$, the S-MCO framework is clearly performing very poorly for an underestimated order of 1. However, the situation improves when using an order of 2 for $\text{Optim}_{\text{PLS}}$ and $\text{Optim}_{\text{MLS}}$, which indicates that for such difficult problems, a sufficiently large but non-exact Walsh order still allows for a reasonable approximation.

From this set of observations, we can draw two general conclusions. First, the two local search optimizers $\text{Optim}_{\text{MLS}}$ and $\text{Optim}_{\text{PLS}}$ are efficient in dealing with the discrete Walsh surrogate, which is to contrast with the $\text{Optim}_{\text{MOEA/D}}$ optimizer. Second, we confirm that an effective surrogate optimizer alone is not able to provide a good performance depending on the available budget and the Walsh order setting. The impact of the order setting is studied next, hence addressing the last component of the S-MCO framework.

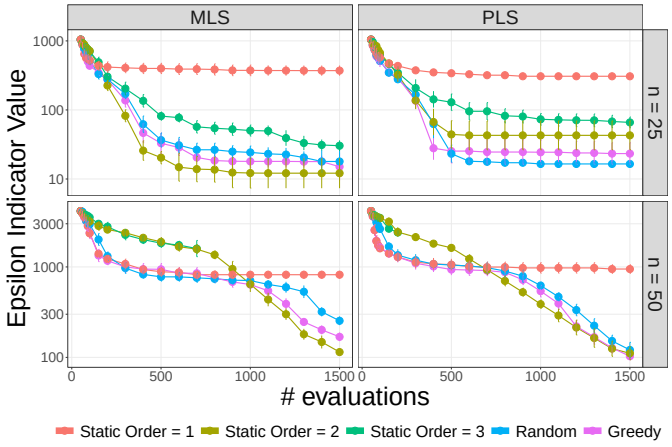

Figure 4. Convergence profile of S-MCO when using different Walsh order settings with the $\text{Select}_{\text{BI}_{\text{norm}}}$ strategy on MUBQP instances.

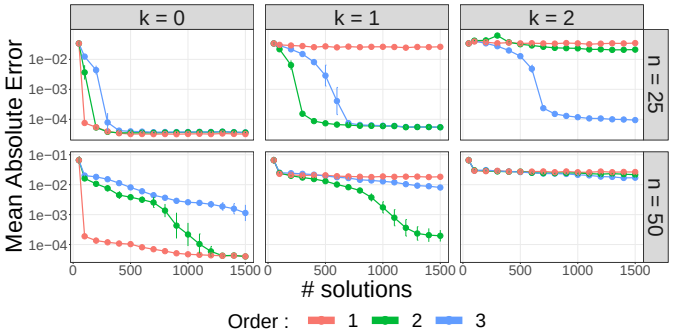

Figure 5. Mean absolute error (MAE) obtained by the Walsh surrogate according to the number of solutions in the training dataset with the $\text{Optim}_{\text{MLS}}$ optimizer combined to the $\text{Select}_{\text{BI}_{\text{norm}}}$ strategy for orders $d \in \{1, 2, 3\}$ on MNK-landscapes.

C. Impact of the Walsh Order

In the following, we study the impact of the Walsh order setting through the analysis of three simple strategies proposed in this paper. We only show the results obtained with the $\text{Optim}_{\text{MLS}}$ and $\text{Optim}_{\text{PLS}}$ surrogate optimizers, given that they were found to substantially outperform the $\text{Optim}_{\text{MOEA/D}}$ optimizer.

From Figures 3 and 4, respectively for MNK-landscapes and MUBQP instances, it becomes now very clear that underestimating the order value (relatively to the exact Walsh transform) has a dramatic impact on the overall performance of the S-MCO framework. Besides, overestimating the order value does not always allow for competitive results. This might seem surprising at first sight, since one may think that overestimating the order can only make the Walsh model more complex without decreasing its accuracy. While such a claim sounds true in theory, increasing the Walsh order makes the model much more challenging to fit and to optimize accurately in practice. This is illustrated in Fig. 5 showing the mean absolute error of the Walsh model as a function of the number of solutions used for training, when executing the S-MCO framework with $\text{Optim}_{\text{MLS}}$ and $\text{Select}_{\text{BI}_{\text{norm}}}$. We clearly see that overestimating the order can lead to a worst fitting quality depending on the characteristic on the degree of non-linearity of the problem and the size of the training set. This is to be attributed to the fact that the more complex the model is, the largest the training data should be in order for the Lasso regression adopted in our experiments to be successful in finding a good fit.

Furthermore, in comparison to a static setting of the Walsh order, the two other dynamic strategies clearly provide a better choice. In fact, they are found to be very competitive, although there is no clear separation between the greedy and the random strategy. In particular, the greedy strategy is not better than the baseline random strategy independently of the surrogate optimizer, problem instance and budget. For instance, we can see that the greedy strategy outperforms the random strategy for MUBQP instances of size $n = 50$. However, this is no more true for $n = 25$ and the $\text{Optim}_{\text{PLS}}$ surrogate optimizer. Similarly, the greedy strategy obtains a better convergence profile than the random strategy when using $\text{Optim}_{\text{PLS}}$ for a MNK-landscapes with $n = 50$ and $k = 0$, whereas this is no more true when combining the greedy strategy with $\text{Optim}_{\text{MLS}}$.

To summarize, we conclude that both dynamic strategies for setting the Walsh order obtain very competitive results. A dynamic order strategy is always better than an underestimated static order, and is better than an overestimated static order in most cases. Surprisingly, it is even better than a static strategy using the exact Walsh transform order for some instances; e.g. MNK-landscapes with $n = 50$ and $k = 1$. These results suggest that more sophisticated dynamic strategies for setting the Walsh order are worth to be investigated in the future.

D. S-MCO *vs. Surrogate-less Approaches*

We conclude our analysis by reporting the performance of the surrogate-assisted S-MCO framework in comparison to a surrogate-less approach. For this purpose, we run the three considered algorithms MOEA/D, PLS, and MLS independently of the S-MCO framework; see Section IV-B. Our results are shown in Figure 6 when using the S-MCO framework configured with the $\text{Select}_{\text{BI}_{\text{norm}}}$ strategy and the greedy order strategy, as these two strategies were found to be a good choice for the S-MCO framework over all experimented instances. We clearly see that all the considered S-MCO variants obtain substantially better approximations, independently of the considered optimizer, instances and budget. This shows the accuracy of the S-MCO framework when configured with an accurate selection strategy and Walsh order setting.

VI. Conclusion

In this paper, we proposed a study on the design of a modular surrogate-assisted framework for expensive multiobjective combinatorial optimization. To the best of our knowledge, this constitutes the first comprehensive investigation in this line. Through an extensive empirical analysis, we provided a systematic investigation of the combined effects of different design components. In particular, we found that there is a non trivial interaction between the strategy allowing to optimize the surrogates used as substitute of the objectives, and the strategy used for selecting the next solution to evaluate. Overall,

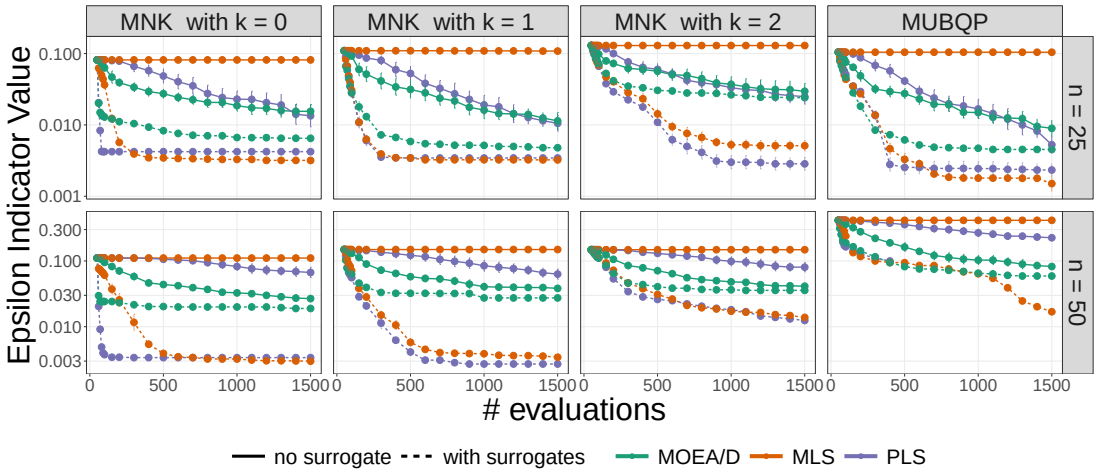

Figure 6. Comparison of surrogate-less approaches with S-MCO using a greedy Walsh order ($\text{Order}_{\text{greedy}}$) and the best-improvement-norm selection strategy ($\text{Select}_{\text{BI}_{\text{norm}}}$).

we show that a selection strategy based on the predicted improvements of candidate solutions with respect to a set of decomposed single-objective sub-problems is highly effective, while local search optimizers are more accurate to deal with the inner optimization of the discrete Walsh surrogate. Finally, we highlighted the importance of the Walsh order setting, and we studied simple dynamic strategies that were shown to accommodate a range of black-box optimization functions with different degrees of landscape difficulty.

Although the proposed S-MCO framework is shown to be extremely effective compared to surrogate-less multiobjective evolutionary algorithms, there remain a number of research questions that are left open. For instance, it would be interesting to leverage the S-MCO framework to support selection strategies based on other multiobjective search paradigms. In fact, our framework is based on decomposition to structure the population of evaluated solutions, and to estimate which new offspring is the most promising for a true evaluation. Another challenging issue is to address other discrete optimization domains, such as permutation problems, for which other single-objective surrogates exist in the specialized literature. The main question is then to study at what extent the proposed S-MCO framework can work efficiently when targeting such settings. Notice that, in principle, the S-MCO framework should work with any accurate single-objective surrogate model. However, investigating the combined effects of its components can only be a function of a target optimization domain/problem. It is our hope that the comprehensive study conducted in this paper will accelerate the establishment of a unified methodology for designing and analyzing surrogate-assisted multiobjective optimization algorithms.

Acknowledgment

This work was supported by the French national research agency (ANR-16-CE23-0013-01) and the Research Grants Council of Hong Kong (RGC Project No. A-CityU101/16).

References

[1] N. Berveglieri, B. Derbel, A. Liefooghe, H. Aguirre, and K. Tanaka, "Surrogate-assisted multiobjective optimization based on decomposition: A comprehensive comparative analysis," in *Proceedings of the Genetic and Evolutionary Computation Conference*, ser. GECCO '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 507–515. [Online]. Available: https://doi.org/10.1145/3321707.3321836

[2] T. Chugh, Y. Jin, K. Miettinen, J. Hakanen, and K. Sindhya, "A surrogate-assisted reference vector guided evolutionary algorithm for computationally expensive many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 1, pp. 129–142, 2018.

[3] I. Loshchilov, M. Schoenauer, and M. Sebag, "A mono surrogate for multiobjective optimization," in *Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation*, ser. GECCO '10. New York, NY, USA: Association for Computing Machinery, 2010, p. 471–478. [Online]. Available: https://doi.org/10.1145/1830483.1830571

[4] L. M. Pavelski, M. R. Delgado, C. P. De Almeida, R. A. Gonçalves, and S. M. Venske, "Elmoea/d-de: Extreme learning surrogate models in multi-objective optimization based on decomposition and differential evolution," in *2014 Brazilian Conference on Intelligent Systems*, 2014, pp. 318–323.

[5] S. Zapotecas Martínez and C. A. Coello Coello, "Moea/d assisted by rbf networks for expensive multi-objective optimization problems," in *Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation*, ser. GECCO '13. New York, NY, USA: Association for Computing Machinery, 2013, p. 1405–1412. [Online]. Available: https://doi.org/10.1145/2463372.2465805

[6] A. Moraglio and A. Kattan, "Geometric generalisation of surrogate model based optimisation to combinatorial spaces," in *Evolutionary Computation in Combinatorial Optimization*, P. Merz and J.-K. Hao, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 142–154.

[7] M. Zaefferer, J. Stork, M. Friese, A. Fischbach, B. Naujoks, and T. Bartz-Beielstein, "Efficient global optimization for combinatorial problems," in *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation*, 2014, p. 871–878.

[8] R. Baptista and M. Poloczek, "Bayesian optimization of combinatorial structures," 2018.

[9] S. Verel, B. Derbel, A. Liefooghe, H. Aguirre, and K. Tanaka, "A Surrogate Model Based on Walsh Decomposition for Pseudo-Boolean Functions," in *Parallel Problem Solving from Nature PPSN XV*, A. Auger, C. M. Fonseca, N. Lourenço, P. Machado, L. Paquete, and D. Whitley, Eds. Cham: Springer International Publishing, 2018, vol. 11102, pp. 181–193.

[10] G. Pruvost, B. Derbel, A. Liefooghe, S. Verel, and Q. Zhang, "Surrogate-assisted multi-objective combinatorial optimization based on decomposition and walsh basis," in *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*, ser. GECCO '20. New York, NY, USA: ACM, 2020, p. 542–550. [Online]. Available: https://doi.org/10.1145/3377930.3390149

[11] F. Leprêtre, S. Verel, C. Fonlupt, and V. Marion, "Walsh functions as surrogate model for pseudo-boolean optimization problems," in *Proceedings of the Genetic and Evolutionary Computation Conference on - GECCO '19*. Prague, Czech Republic: ACM Press, 2019, pp. 303–311.

[12] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 712–731, 2007.

[13] L. Paquete and T. Stützle, "A study of stochastic local search algorithms for the biobjective QAP with correlated flow matrices," *European Journal of Operational Research*, vol. 169, no. 3, pp. 943–959, 2006.

[14] K. Sindhya, K. Deb, and K. Miettinen, "A local search based evolutionary multi-objective optimization approach for fast and accurate convergence," in *Parallel Problem Solving from Nature – PPSN X*, G. Rudolph, T. Jansen, N. Beume, S. Lucas, and C. Poloni, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 815–824.

[15] M. Ehrgott, *Multicriteria optimization*, 2nd ed. Springer, 2005.

[16] K. Deb, *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, 2001.

[17] C. A. Coello Coello, G. B. Lamont, and D. A. Van Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems*, 2nd ed. Springer, 2007.

[18] A. Trivedi, D. Srinivasan, K. Sanyal, and A. Ghosh, "A survey of multiobjective evolutionary algorithms based on decomposition," *IEEE TEC*, vol. 21, no. 3, pp. 440–462, 2017.

[19] B. Derbel, A. Liefooghe, Q. Zhang, S. Verel, H. Aguirre, and K. Tanaka, "A set-oriented MOEA/D," in *GECCO 2018 - Genetic and Evolutionary Computation Conference*. Kyoto, Japan: ACM Press, Jul. 2018, pp. 617–624.

[20] Y. Jin, "Surrogate-assisted evolutionary computation: Recent advances and future challenges," *Swarm and Evolutionary Computation*, vol. 1, no. 2, pp. 61–70, 2011.

[21] T. Bartz-Beielstein and M. Zaefferer, "Model-based methods for continuous and discrete global optimization," *Applied Soft Computing*, vol. 55, pp. 154 – 167, 2017.

[22] J. L. Walsh, "A Closed Set of Normal Orthogonal Functions," *American Journal of Mathematics*, vol. 45, no. 1, p. 5, 1923.

[23] A. D. Bethke, "Genetic algorithms as function optimizers," Ph.D. dissertation, University of Michigan, 1980.

[24] T. Hastie, R. Tibshirani, and M. Wainwright, *Statistical Learning with Sparsity: The Lasso and Generalizations*, 1st ed. Chapman and Hall/CRC, 2015.

[25] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.

[26] H. Aguirre and K. Tanaka, "Working principles, behavior, and performance of MOEAs on MNK-landscapes," *European Journal of Operational Research*, vol. 181, no. 3, pp. 1670–1690, 2007.

[27] S. Verel, A. Liefooghe, L. Jourdan, and C. Dhaenens, "On the structure of multiobjective combinatorial search space: MNK-landscapes with correlated objectives," *European Journal of Operational Research*, vol. 227, no. 2, pp. 331–342, 2013.

[28] A. Liefooghe, S. Verel, and J.-K. Hao, "A hybrid metaheuristic for multiobjective unconstrained binary quadratic programming," *Applied Soft Computing*, vol. 16, pp. 10–19, 2014.

[29] G. Kochenberger, J.-K. Hao, F. Glover, M. Lewis, Z. Lü, H. Wang, and Y. Wang, "The unconstrained binary quadratic programming problem: a survey," *Journal of Combinatorial Optimization*, vol. 28, no. 1, pp. 58–81, 2014.

[30] S. A. Kauffman, *The Origins of Order*. Oxford University Press, 1993.

[31] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. Grunert da Fonseca, "Performance assessment of multiobjective optimizers: An analysis and review," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 117–132, 2003.