Mathematical and Computational Applications

MDPI

Review

Surrogate Modeling Approaches for Multiobjective Optimization: Methods, Taxonomy, and Results

Kalyanmoy Deb *,†, Proteek Chandan Roy † and Rayan Hussein †

Computational Optimization and Innovation (COIN) Laboratory, Michigan State University, East Lansing, MI 48824, USA; royproteekchandan@gmail.com (P.C.R.); husseinr@egr.msu.edu (R.H.)
* Correspondence: kdeb@egr.msu.edu
† These authors contributed equally to this work.

Abstract: Most practical optimization problems are comprised of multiple conflicting objectives and constraints which involve time-consuming simulations. Construction of metamodels of objectives and constraints from a few high-fidelity solutions and a subsequent optimization of metamodels to find in-fill solutions in an iterative manner remain a common metamodeling based optimization strategy. The authors have previously proposed a taxonomy of 10 different metamodeling frameworks for multiobjective optimization problems, each of which constructs metamodels of objectives and constraints independently or in an aggregated manner. Of the 10 frameworks, five follow a generative approach in which a single Pareto-optimal solution is found at a time and other five frameworks were proposed to find multiple Pareto-optimal solutions simultaneously. Of the 10 frameworks, two frameworks (M3-2 and M4-2) are detailed here for the first time involving multimodal optimization methods. In this paper, we also propose an adaptive switching based metamodeling (ASM) approach by switching among all 10 frameworks in successive epochs using a statistical comparison of metamodeling accuracy of all 10 frameworks. On 18 problems from three to five objectives, the ASM approach performs better than the individual frameworks alone. Finally, the ASM approach is compared with three other recently proposed multiobjective metamodeling methods and superior performance of the ASM approach is observed. With growing interest in metamodeling approaches for multiobjective optimization, this paper evaluates existing strategies and proposes a viable adaptive strategy by portraying importance of using an ensemble of metamodeling frameworks for a more reliable multiobjective optimization for a limited budget of solution evaluations.

Keywords: surrogate modeling; multiobjective optimization; evolutionary algorithms; kriging method; ensemble method; adaptive algorithm

Citation: Deb, K.; Roy, P.C.; Hussein, R. Surrogate Modeling Approaches for Multiobjective Optimization: Methods, Taxonomy, and Results. *Math. Comput. Appl.* **2021**, *26*, 5. https://doi.org/10.3390/mca26010005

Received: 25 October 2020
Accepted: 27 December 2020
Published: 31 December 2020

1. Introduction

Practical problems often require expensive simulation of accurate high-fidelity models. To get close to the optimum of these models, most multiobjective optimization algorithms need to compute a large number of solution evaluations. However, in practice, only a handful of solution evaluations are allowed due to the overall time constraint available to solve such problems. Researchers usually resort to surrogate models or metamodels constructed from a few high-fidelity solution evaluations to replace computationally expensive models to drive an optimization task [1–3]. For example, Gaussian process model, Kriging, or response surface method is commonly used. The Kriging method is of particular interest, since it is able to provide an approximated function as well as an estimate of uncertainty of the prediction of the function [4].

In extending the metamodeling concept to multiobjective optimization problems, an obvious issue is that multiple objective and constraint functions are required to be metamodeled before proceeding with the optimization algorithm. Despite this challenge of multiple metamodeling efforts, a good number of studies have been made to solve

computationally expensive multiobjective optimization problems using metamodeling based evolutionary algorithms [5–10]. However, most of these studies ignored constraints and extending an unconstrained optimization algorithm to constrained optimization is not trivial [11]. In any case, the structure of most of these methods is as follows. Starting an initial archive of solutions obtained by an usual Latin-hypercube sampling, a metamodel for each objective and constraint function is built independently [12,13]. Then, in an *epoch*–one cycle of metamodel development and their use to obtain a set of *in-fill* solutions, an evolutionary multiobjective optimization (EMO) algorithm is used to optimize the metamodeled objectives and constraints to find one or more in-fill points. Thereafter, the in-fill points are evaluated using high-fidelity models and saved into the archive. Next, new metamodels are built using the augmented archive members and the procedure is repeated in several epochs until the allocated number of solution evaluations is consumed [5,14–19]. Many computationally expensive optimization problems involve noisy high-fidelity simulation models. Noise can come from inputs, stochastic processes of the simulation, or the output measurements. In this paper, we do not explicitly discuss the effect of noise in handling metamodeling problems, but we recognize that this is an important matter in solving practical problems.

In a recent taxonomy study [20], authors have categorized different plausible multiobjective metamodeling approaches into 10 frameworks, of which the above-described popular method falls within the first two frameworks—M1-1 or M1-2, depending on whether a single or multiple nondominated in-fill solutions are found in each epoch. The other eight frameworks were not straightforward from a point of view extending single-objective metamodeling approaches to multiobjective optimization and hence were not explored in the past. Moreover, the final two frameworks (M5 and M6) attempt to metamodel an EMO algorithm's implicit overall fitness (or selection) function directly, instead of metamodeling an aggregate or individual objective and constraint functions. There is an advantage of formulating a taxonomy, so that any foreseeable future metamodeling method can also be categorized to fall within one of the 10 frameworks. Moreover, the taxonomy also provides new insights to other currently unexplored ways of handling metamodels within a multiobjective optimization algorithm.

So far, each framework has been applied alone in one complete optimization run to solve a problem, but in a recent study [21], a manual switching of one framework to another after 50% of allocated solution evaluations has produced improved results. An optimization process goes through different features of the multiobjective landscape and it is natural that a different metamodeling framework may be efficient at different phases of a run. These studies are the genesis of this current study, in which we propose an adaptive switching based metamodeling (ASM) approach, which automatically finds one of the 10 best-performing frameworks at the end of each epoch after a detailed statistical study, thereby establishing self-adaptive and efficient overall metamodeling based optimization approach.

In the remainder of the paper, Section 2 briefly describes a summary of recent related works. Section 3 provides a brief description of each of 10 metamodeling frameworks for multiobjective optimization. The proposed ASM approach is described in Section 4. Our extensive results on unconstrained and constrained test problems for each framework alone and the ASM approach are presented in Section 5. A comparative study of the ASM approach with three recent existing algorithms is presented in Section 5.5. We summarize our study of the switching framework based surrogate-assisted optimization with future research directions in Section 6.

2. Past Methods of Metamodeling for Multiobjective Optimization

We consider the following original multi- or many-objective optimization problem (P), involving n real-valued variables ($\mathbf{x}$), J inequality constraints ($\mathbf{g}$) (equality constraints, if any, are assumed to be converted to two inequality constraints), and M objective functions ($\mathbf{f}$):

$$\begin{array}{ll} \text{Minimize} & (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_M(\mathbf{x})), \\ \text{Subject to} & g_j(\mathbf{x}) \leq 0, \quad j = 1, 2, \ldots, J, \\ & x_i^{(L)} \leq x_i \leq x_i^{(U)}, \quad i = 1, 2, \ldots, n. \end{array} \tag{1}$$

In this study, we assume that all objective and constraint functions are *computationally expensive* to compute and that they need to be computed independent to each other for every new solution $\mathbf{x}$. To distinguish from the original functions, the respective metamodeled function is represented with a "tilde" (such as, $\widetilde{f}_i(\mathbf{x})$ or $\widetilde{g}_j(\mathbf{x})$). The resulting metamodeled problem is denoted here as MP, which is formed with developed metamodels of individual objective and constraints or their aggregates. *In-fill* solutions are defined as optimal solutions of problem MP. It is assumed here that constructing the metamodels and their comparisons among each other consume comparatively much less time than evaluating objective and constraints exactly, hence, if the metamodels are close to the original functions, the process can end up with a huge savings in computational time without much sacrifice in solution accuracy. Naturally, in-fill solutions (obtained from metamodels) need to be evaluated using original objective and constraints (termed here as "high-fidelity" evaluations) and can be used to refine the metamodels for their subsequent use within the overall optimization approach.

A number of efficient metamodeling frameworks have been proposed recently for multiobjective optimization [10,22–28], including a parallel implementation concept [29]. These frameworks use different metamodeling methods to approximate objective and constraint functions, such as radial basis functions (RBFs), Kriging, Bayesian neural network, support vector regression, and others [30]. Most of these methods proposed a separate metamodel for each objective and constraint function, akin to our framework M1. Another study have used multiple spatially distributed surrogate models for multiobjective optimization [31]. It is clear that this requires a lot of metamodeling efforts and metamodeling errors from different models can accrue and make the overall optimization to be highly error-prone. As will be clear later, these methods will fall under our M1-2 framework.

Zhang et al. [14] proposed the MOEA/D-EGO algorithm which metamodeled each objective function independently. They constructed multiple expected global optimization (EGO) functions for multiple reference lines of the MOEA/D approach to find a number of trade-off solutions in each optimization task. No constraint handling procedure was suggested. Thus, this method falls under our M1-2 framework.

Chugh et al. [23] proposed a surrogate-assisted adaptive reference vectors guided evolutionary algorithm (K-RVEA) for computationally expensive optimization problems with more than three objectives. Since all objectives and constraints are metamodeled separately, this method also falls under our M1-2 framework. While no constraint handling was proposed with the original study, a later version included constraint handling [32].

Zhao et al. [24] classified the sample data into clusters based on their similarities in the variable space. Then, a local metamodel was built for each cluster of the sample data. A global metamodel is then built using these local metamodels considering their contributions in different regions of the variable space. Due to the construction and optimization of multiple metamodels, one for each cluster, this method belongs to our M-3 framework. The use of a global metamodel by combining all local cluster-wise metamodels qualify this method under the M3-2 framework. No constraint handling method is suggested.

Bhattacharjee et al. [25] used an independent metamodel for each objective and constraint using different metamodeling methods: RBF, Kriging, first and second-order re-

sponse surface models, and multilayer perceptrons. NSGA-II method is used to optimized metamodeled version of the problem. Clearly, this method falls under our M1-2 category.

Wang et al. [26] used independent metamodeling of objectives but combined them using a weight-sum approach proposed an ensemble-based model management strategy for surrogate-assisted evolutionary algorithm. Thus, due to modeling a combined objective function, this method falls under our M3-1 framework. A global model management strategy inspired from committee-based active learning (CAL) was developed, searching for the best and most uncertain solutions according to a surrogate ensemble using a particle swarm optimization (PSO) algorithm. In addition, a local surrogate model is built around the best solution obtained so far. Then, a PSO algorithm searches on the local surrogate to find its optimum and evaluates it. The evolutionary search using the global model management strategy switches to the local search once no further improvement can be observed and vice versa.

Pan et al. [33] proposed a classification based surrogate-assisted evolutionary algorithm (CSEA) for solving unconstrained optimization problems by using an artificial neural network (ANN) as a surrogate model. The surrogate model aims to learn the dominance relationship between the candidate solutions and a set of selected reference solutions. Due to a single metamodel to find the dominance structure involving all objective functions, this algorithm falls under our M3-2 framework.

Deepti et al. [34] suggested a reduced and simplified model of each objective function in order to reduce the computational efforts.

Recent studies on nonevolutionary optimization methods for multiobjective optimization using trust-region method [35,36] and using decomposition methods [37] are proposed as well.

A recent study [38] reviewed multiobjective metamodeling approaches and suggested a taxonomy of the existing methods based on whether the surrogate assisted values match well the original function values. Three broad categories were suggested: (i) algorithms that do not use any feedback from the original function values, (ii) algorithms that use a fixed number of feedback, and (iii) algorithms that adaptively decide which metamodeled solutions must be checked with the original function values. This extensive review reported that most existing metamodeling approaches used a specific EMO algorithm—NSGA-II [39]. While a check on the accuracy of a metamodel is important for its subsequent use, this is true for both single and multiobjective optimization and no specific issues related to multiobjective optmization were discussed in the review paper.

Besides the algorithmic developments, a number of studies have applied metamodeling methods to practical problems with a limited budget of solution evaluations [40–47], some restricting to a few hundreds [48].

Despite all the above all-around developments, the ideas that most distinguish surrogate modeling in multiobjective optimization from their single-objective counterparts were not addressed well. They are (i) how to fundamentally handle multiple objectives and constraints either through a separate modeling of each or in an aggregated fashion? and (ii) how to make use of the best of different multiple surrogate modeling approaches adaptively within an algorithm? In 2016, Rayan et al. [5] have proposed a taxonomy in which 10 metamodeling frameworks were proposed to address the first question. This paper addresses the second question in a comprehensive manner using the proposed 10 metamodeling frameworks using an ensemble method.

Ensemble methods have been used in surrogate-assisted optimization for solving expensive problems [49–53], but in most of these methods, an ensemble of different metamodeling methods, such as RBF, Kriging, response surfaces, are considered to choose a single suitable method. While such studies are important, depending on the use of objectives and constraints, each such method will fall in one of the first eight frameworks presented in this paper. No effort is made to consider an ensemble of metamodeling frameworks for combining multiple objectives and constraints differently and choosing the most suitable one for optimization. In this paper, we use an ensemble of 10 metamodeling

frameworks [5,20] described in the next section and propose an adaptive selection scheme of choosing one in an iterative manner thereafter.

3. A Taxonomy for Multiobjective Metamodeling Frameworks

Having M objective and J constraints to be metamodeled, there exist many plausible ways to develop a metamodeling based multiobjective optimization methods. Thus, there is a need to classify different methods into a few finite clusters so that they can be compared and contrasted with each other. Importantly, such a classification or taxonomy study can provide information about methods which are still unexplored. A recently proposed taxonomy study [20] put forward 10 different frameworks based on the metamodeling objective and constraint functions based on their individual or aggregate modeling, as illustrated in Figure 1.

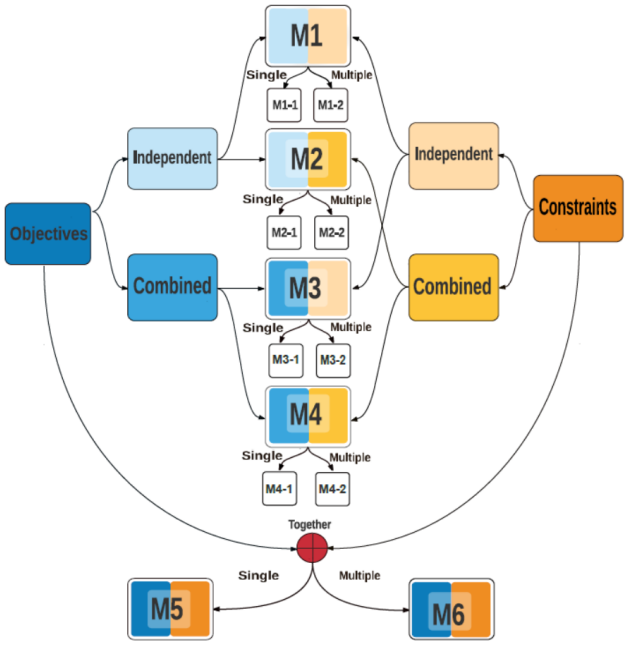


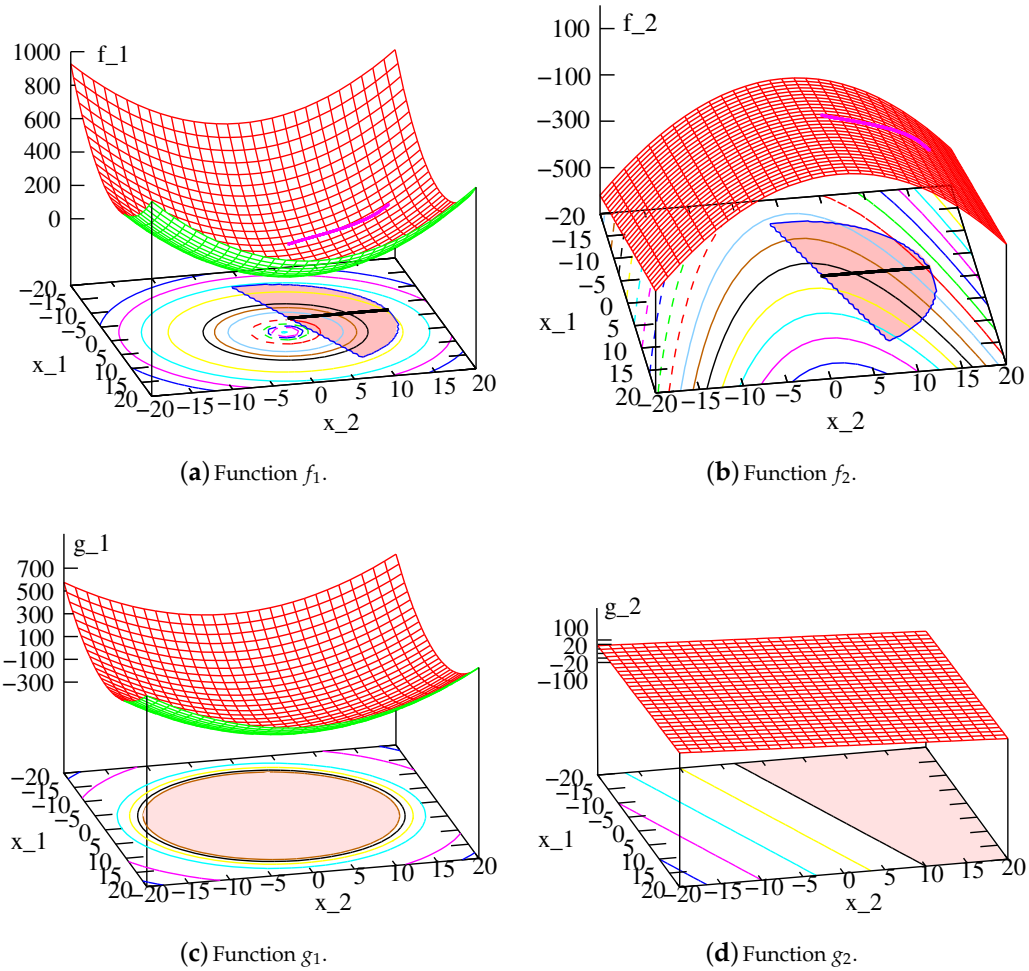
Figure 1. The proposed taxonomy of 10 different metamodeling frameworks for multi- and many-objective optimization. (Taken from [20]).

We believe most ideas of collectively metamodeling all objectives and constraints can be classified into one of these 10 frameworks. We describe each of the 10 frameworks below in details for the first time.

We explain each framework using a two-variable, two-objective SRN problem [54,55] as an example:

$$\begin{array}{ll} \text{Minimize} & f_1(\mathbf{x}) = 2 + (x_1 - 2)^2 + (x_2 - 1)^2, \\ \text{Minimize} & f_2(\mathbf{x}) = 9x_1 - (x_2 - 1)^2, \\ \text{Subject to} & g_1(\mathbf{x}) = x_1^2 + x_2^2 - 225 \leq 0, \\ & g_2(\mathbf{x}) = x_1 - 3x_2 + 10 \leq 0, \\ & -20 \leq (x_1, x_2) \leq 20. \end{array} \tag{2}$$

The PO solutions are known to be as follows: $x_1^* = -2.5$ and $x_2^* \in [2.5, 14.79]$. To apply a metamodeling approach, one simple idea is to metamodel all four functions. The functions and the respective PO solutions are marked on f_1 and f_2 plots shown in Figure 2a,b, respectively. The feasible regions for g_1 anf g_2 are shown in Figure 2c,d.



(**a**) Function f_1. (**b**) Function f_2.

(**c**) Function g_1. (**d**) Function g_2.

Figure 2. Two objectives and two constraints are shown for SRN problem. The combined feasible region is shown in the contour plot. PO solutions lie on the black line marked inside the feasible region.

3.1. M1-1 and M1-2 Frameworks

Most existing multiobjective metamodeling approaches are found to fall in these two frameworks [20]. In M1-1 and M1-2, a total of $(M + J)$ metamodels (M objectives and J constraints) are constructed. The metamodeling algorithm for M1-1 and M1-2 starts with an archive of initial population ($\mathcal{A}_0$ of size N_0) created using the Latin hypercube sampling (LHS) method on the entire search space, or by using any other heuristics of the problem. Each objective function ($f_i(\mathbf{x})$, for $i = 1, \ldots, M$) is first normalized to obtain a normalized function $\underline{f}_i(\mathbf{x})$ using the minimum ($f_i^{\min}$) and maximum ($f_i^{\max}$) values of all high-fidelity evaluation of archive members, so that the minimum and maximum values of $\underline{f}_i(\mathbf{x})$ is zero and one, respectively:

$$\underline{f}_i(\mathbf{x}) = \frac{f_i(\mathbf{x}) - f_i^{\min}}{f_i^{\max} - f_i^{\min}}. \tag{3}$$

Then, metamodels are constructed for each of the M normalized objective functions independently: $(\widetilde{\underline{f}}_1(\mathbf{x}), \ldots, \widetilde{\underline{f}}_M(\mathbf{x})), \forall i \in \{1, 2, \ldots, M\}$ using a chosen metamodeling method. For all implementations here, we use the Kriging metamodeling method [56] for all frameworks of this study.

Each constraint function ($g_j(\mathbf{x})$, for $j = 1, \ldots, J$) is first normalized to obtain a normalized constraint function ($\underline{g}_j(\mathbf{x})$) using standard methods [57], and then metamodeled

separately to obtain an approximate function ($\widetilde{g}_j(\mathbf{x})$) using the same metamodeling method (Kriging method is adopted here) used for metamodeling objective functions.

In M1-1, all metamodeled normalized objectives are combined into a single *aggregated* function and optimized with all separately metamodeled constraints to find a *single* in-fill point using a single-objective evolutionary optimization algorithm (real-coded genetic algorithm (RGA) [54] is used here). In τ generations of RGA (defining an epoch), the following achievement scalarization aggregation function ($\text{ASF}_{12}(\mathbf{x}, \mathbf{z})$) [58] is optimized for every $\mathbf{z}$ vector:

$$\text{Problem O1-1: Solution: } \mathbf{x}^*(\mathbf{z}), \quad \begin{cases} \text{Minimize} & \text{ASF}_{12}(\mathbf{x}, \mathbf{z}) = \max_{j=1}^{M} \left(\widetilde{f}_j(\mathbf{x}) - z_j \right), \\ \text{Subject to} & \widetilde{g}_j(\mathbf{x}) \leq 0, \quad j = 1, 2, \ldots, J, \\ & x_i^{(L)} \leq x_i \leq x_i^{(U)}, \quad i = 1, 2, \ldots, n, \end{cases} \tag{4}$$

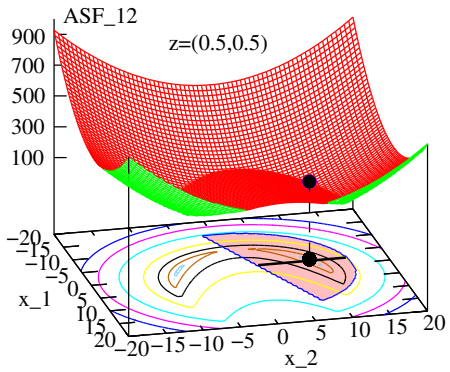
where the vector $\mathbf{z}$ is one of the Das and Dennis's [59] point on the unit simplex on the M-dimensional hyperspace (making $\sum_{j=1}^{M} z_i = 1$). Thus, for each of H different $\mathbf{z}$ vectors, one optimization problem (O1-1) is formed with an equi-angled weight vector, and solved one at time to find a total of H in-fill solutions using a real-parameter genetic algorithm (RGA). Figure 3a shows the infill solution for $\mathbf{z} = (0.5, 0.5)$ for the SRN problem. Notice, the ASF_{12} function constitutes a minimum point on the Pareto-optimal (PO) line (black line on the contour plot) for the specific $\mathbf{z}$-vector. If the exact ASF_{12} function can be constructed as a metamodeled function from a few high-fidelity evaluations, one epoch would be enough to find a representative PO set. However, since the metamodeled function is expected to have a difference from the original function, several epochs will be necessary to get close to the true PO set. For a different $\mathbf{z}$-vector, the ASF_{12} function will have a different optimal solution, but it will fall on the PO line. The ASF_{12} model, constructed from metamodeled objective and constraint functions, will produce optimal solutions on the Pareto set for different $\mathbf{z}$-vectors. Multiple applications of a RGA will discover a well-distributed set of multiple in-fill points one at a time.

The RGA procedure uses a *trust-region* concept, which we describe in detail in Section 4.3. The best solution for each $\mathbf{z}$ is sent for a high-fidelity evaluation. The solution is then included in the archive ($\mathcal{A}_1$) of all high-fidelity solutions. After all H solutions are included in the archive, one epoch of the M1-1 framework optimization problem is considered complete. In the next epoch, all high-fidelity solutions are used to normalize and metamodel all $(M + J)$ objective functions and constraints, and the above process is repeated to obtain $\mathcal{A}_2$. The process is continued until all prespecified maximum solution evaluations ($\text{SE}_{\max}$) is completed. Nondominated solutions of final archive $\mathcal{A}_t$ is declared as outcome of the whole multiobjective surrogate-assisted approach.

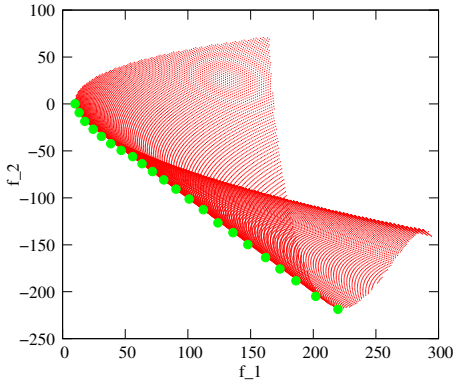
In M1-2, the following M-objective optimization problem,

$$\text{Problem O1-2: Solutions: } \mathbf{x}^{i,*},\ i = 1, \ldots, H \quad \begin{cases} \text{Minimize} & \left(\widetilde{f}_1(\mathbf{x}), \widetilde{f}_2(\mathbf{x}), \ldots, \widetilde{f}_M(\mathbf{x}) \right), \\ \text{Subject to} & \widetilde{g}_j(\mathbf{x}) \leq 0, \quad j = 1, 2, \ldots, J, \\ & x_i^{(L)} \leq x_i \leq x_i^{(U)}, \quad i = 1, 2, \ldots, n, \end{cases} \tag{5}$$

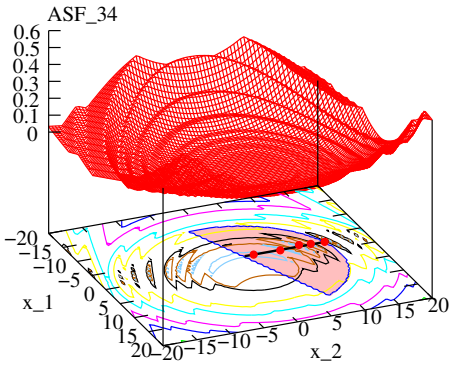
constructing $(M + J)$ metamodels in each epoch, is solved to find H in-fill solutions in a *single run* with an EMO/EMaO procedure. We use NSGA-II procedure [39] for two-objective problems, and NSGA-III [60] for three or more objective problems here. All H solutions are then evaluated using high-fidelity models and are included in the archive for another round of metamodel construction and optimization for the next epoch. The process is continued until $\text{SE}_{\max}$ evaluations are done. Figure 3b shows that when NSGA-II optimizes a well-approximated metamodel to the original problem, the obtained solutions will lie on the true PO front.



(a) The ASF solution with $\mathbf{z} = (0.5, 0.5)$ in M1-1, M2-1, M3-1 and M4-1. The solution lies on the true PO set (black line).



(b) Efficient solutions in M1-2 and M2-2. For illustration, a few PO solutions are shown in green circles.



(c) minASF_{34} in M3-2 and M4-2. For illustration, five $\mathbf{z}$-vectors and their respective PO solutions, found simultaneously, are shown in red circles.

Figure 3. In-fill solutions for different frameworks for SRN problem. True functions are plotted here, however, different metamodeling frameworks use different approximations to find in-fill solutions on the true PO set.

3.2. Frameworks M2-1 and M2-2

For M2-1 and M2-2, a single aggregated constraint violation function (ACV($\mathbf{x}$)) is first constructed using the normalized constraint functions ($\underline{g}_j(\mathbf{x})$, $j = 1, \ldots, J$) at high-fidelity solutions from the archive ($\mathbf{x} \in \mathcal{A}_t$), as follows:

$$\text{ACV}(\mathbf{x}) = \begin{cases} \sum_{j=1}^{J} \underline{g}_j(\mathbf{x}), & \text{if } \mathbf{x} \text{ is feasible,} \\ \sum_{j=1}^{J} \langle \underline{g}_j(\mathbf{x}) \rangle, & \text{otherwise,} \end{cases} \tag{6}$$

where the bracket operator $\langle \alpha \rangle$ is α, if $\alpha > 0$; and zero, otherwise. It is clear from the above equation that for high-fidelity solutions, ACV($\mathbf{x}$) takes a negative value for feasible solutions and a positive value for an infeasible solution. In M2-1 and M2-2, the constraint violation function (ACV($\mathbf{x}$)) is then metamodeled to obtain $\widetilde{\text{ACV}}(\mathbf{x})$, instead of every constraint function ($g_j(\mathbf{x})$) metamodeled in M1-1 and M1-2. This requires a total of $(M+1)$ metamodel constructions (M objectives and one constraint violation function) at each epoch. In M2-1, the following problem

$$\text{Problem O2-1:} \quad \text{Solution: } \mathbf{x}^*(\mathbf{z}) \quad \begin{cases} \text{Minimize} & \text{ASF}_{12}(\mathbf{x}, \mathbf{z}) = \max_{j=1}^{M} \left(\widetilde{\underline{f}}_j(\mathbf{x}) - z_j \right), \\ \text{Subject to} & \widetilde{\text{ACV}}(\mathbf{x}) \leq 0, \\ & x_i^{(L)} \leq x_i \leq x_i^{(U)}, \quad i = 1, 2, \ldots, n, \end{cases} \tag{7}$$

is solved to find one in-fill point for each reference line originating from one of the chosen Das-Dennis reference points $\mathbf{z}$. Similarly, M2-2 solves the following problem:

$$\text{Problem O2-2: Solutions: } \mathbf{x}^{i,*},\ i=1,\ldots,H \quad \begin{cases} \text{Minimize} & \left(\widetilde{\underline{f}}_1(\mathbf{x}), \widetilde{\underline{f}}_2(\mathbf{x}), \ldots, \widetilde{\underline{f}}_M(\mathbf{x})\right), \\ \text{Subject to} & \widetilde{\text{ACV}}(\mathbf{x}) \leq 0, \\ & x_i^{(L)} \leq x_i \leq x_i^{(U)}, \quad i=1,2,\ldots,n, \end{cases} \tag{8}$$

to find H in-fill solutions simultaneously. The rest of the M2-1 and M2-2 procedures are identical to that in M1-1 and M1-2, respectively. RGA is used to solve each optimization problem in M2-1 to find one solution at a time, and NSGA-II or NSGA-III is used in M2-2 depending on number of objectives in the problem. Thus, M2-1 requires an archive to store each solution, whereas M2-2 does not require an archive.

3.3. M3-1 and M3-2 Frameworks

In these two methods, instead of metamodeling each normalized objective function $\underline{f}_i(\mathbf{x})$ for $i=1,\ldots,M$ independently, we first aggregate them to form the following ASF_{34} function for each high-fidelity solution $\mathbf{x}$:

$$\text{ASF}_{34}(\mathbf{x},\mathbf{z}) = \max_{j=1}^{M}\left(\underline{f}_j(\mathbf{x}) - z_j\right), \tag{9}$$

where $\mathbf{z}$ is defined as before. Note this formulation is different from ASF_{12} in that the ASF formulation is made with the original normalized objective functions $\underline{f}_j$ here. Then, one ASF_{34} function (for a specific $\mathbf{z}$-vector) is metamodeled to obtain $\widetilde{\text{ASF}}_{34}(\mathbf{x},\mathbf{z})$, along with J separate metamodels for J constraints ($\widetilde{\underline{g}}_j$) to solve the following problem for M3-1:

$$\text{Problem O3-1: Solutions: } \mathbf{x}^*(\mathbf{z}) \quad \begin{cases} \text{Minimize} & \widetilde{\text{ASF}}_{34}(\mathbf{x},\mathbf{z}), \\ \text{Subject to} & \widetilde{\underline{g}}_j(\mathbf{x}) \leq 0, \quad j=1,2,\ldots,J, \\ & x_i^{(L)} \leq x_i \leq x_i^{(U)}, \quad i=1,2,\ldots,n. \end{cases} \tag{10}$$

For every $\mathbf{z}$, a new in-fill point is found by solving the above problem using the same RGA, discussed for M1-1. Every in-fill point is stored in an archive to compare with M$*$-2 methods, which creates multiple solutions in one run, thereby not requiring an explicit archive. In M3-2, the following problem is solved:

$$\text{Problem O3-2: Solutions: } \mathbf{x}^{i,*},\ i=1,\ldots,H \quad \begin{cases} \text{Minimize} & \text{minASF}_{34}(\mathbf{x}) = \min_{\mathbf{z}} \widetilde{\text{ASF}}_{34}(\mathbf{x},\mathbf{z}), \\ \text{Subject to} & \widetilde{\underline{g}}_j(\mathbf{x}) \leq 0, \quad j=1,2,\ldots,J, \\ & x_i^{(L)} \leq x_i \leq x_i^{(U)}, \quad i=1,2,\ldots,n, \end{cases} \tag{11}$$

in which the objective function of $\mathbf{x}$ is computed as the minimum $\widetilde{\text{ASF}}_{34}$ for all $\mathbf{z}$-vectors at $\mathbf{x}$. Figure 3c shows the multimodal objective function $\text{minASF}_{34}(\mathbf{x})$ for the SRN problem, clearly indicating multiple local optima on the PO front. Notice how the minASF_{34} function has ridges and creates multiple optima on the PO set, one for each reference line. Due to the complexity involved in this function, it is clear that a large number of high-fidelity points will be necessary to make a suitable metamodel with a high accuracy. Besides the need of more points, there is another issue that needs a discussion. Both M3-1 and M3-2 requires H, $\widetilde{\text{ASF}}_{34}(\mathbf{x},\mathbf{z})$ and J constraint functions to be metamodeled, thereby making a total of $(H+J)$ metamodels in each epoch. Since each of multiple optima of the minASF_{34} function will finally lead us to a set of PO solutions, we would need an efficient multimodal optimization algorithm, instead of a RGA, to solve the metamodeled minASF_{34} function.

We use a *multimodal* single-objective evolutionary algorithm to find H multimodal in-fill points of minASF_{34} simultaneously. We propose a multimodal RGA (or MM-RGA) which starts with a random population of size N for this purpose. In each generation, the

population (P_t) is modified to a new population (P_{t+1}) by using selection, recombination, and mutation operators. The selection operator emphasizes multiple diverse solutions as follows. First, a *fitness* is assigned to each population member $\mathbf{x}$ by computing $\widetilde{\text{ASF}}_{34}(\mathbf{x}, \mathbf{z})$ for all H, $\mathbf{z}$-vectors and then assigning the smallest value as the fitness. Then, we apply the binary tournament selection to choose a parent using the following selection function:

$$\text{SF}(\mathbf{x}) = \begin{cases} \text{minASF}_{34}(\mathbf{x}), & \text{if } \mathbf{x} \text{ is feasible,} \\ \text{minASF}_{34}^{\max} + \sum_{j=1}^{J} \langle \widetilde{\underline{g}}_j(\mathbf{x}) \rangle, & \text{otherwise,} \end{cases} \tag{12}$$

where $\text{minASF}_{34}^{\max}$ is the maximum $\text{minASF}_{34}(\mathbf{x})$ value of all feasible population members of MM-RGA. The above selection function has the following effects. If two solutions are feasible, $\text{minASF}_{34}(\mathbf{x})$ is used to select the winner. If one is feasible and the other is infeasible, the former is chosen, and for two infeasible members, the one with smaller constraint violation $\sum_{j=1}^{J} \langle \widetilde{\underline{g}}_j(\mathbf{x}) \rangle$ is chosen. After N offspring population members are thus created, we merge the population to form a combined population of $2N$ members. The best solution to each $\mathbf{z}$-vector is then copied to P_{t+1}. In the event of a duplicate, the second best solution for the $\mathbf{z}$-vector is chosen. If H is smaller than N, then the process is repeated to select a second population member for as many $\mathbf{z}$-vectors as possible. Thus, at the end of the MM-RGA procedure, exactly H in-fill solutions are obtained.

3.4. Frameworks M4-1 and M4-2

In these two frameworks, constraints are first combined to a single constraint violation function $\text{ACV}(\mathbf{x})$ as in M2-1 (Equation (6)) and then ACV is metamodeled to obtain $\widetilde{\text{ACV}}(\mathbf{x})$. The following problem is then solved:

$$\text{Problem O4-1: Solution: } \mathbf{x}^*(\mathbf{z}) \quad \begin{cases} \text{Minimize} & \widetilde{\text{ASF}}_{34}(\mathbf{x}, \mathbf{z}), \\ \text{Subject to} & \widetilde{\text{ACV}}(\mathbf{x}) \leq 0, \\ & x_i^{(L)} \leq x_i \leq x_i^{(U)}, \quad i = 1, 2, \ldots, n, \end{cases} \tag{13}$$

to find a single in-fill solution for every $\mathbf{z}$. An archive is built with in-fill solutions. In M4-2, following problem is solved to find H in-fill solutions simultaneously:

$$\text{Problem O4-2: Solutions: } \mathbf{x}^{i,*},\ i = 1, \ldots, H \quad \begin{cases} \text{Minimize} & \text{minASF}_{34}(\mathbf{x}) = \min_{\mathbf{z}} \widetilde{\text{ASF}}_{34}(\mathbf{x}, \mathbf{z}), \\ \text{Subject to} & \widetilde{\text{ACV}}(\mathbf{x}) \leq 0, \\ & x_i^{(L)} \leq x_i \leq x_i^{(U)}, \quad i = 1, 2, \ldots, n, \end{cases} \tag{14}$$

Both these frameworks require H, $\widetilde{\text{ASF}}_{34}(\mathbf{x}, \mathbf{z})$ and one ACV function to be metamodeled, thereby making a total of $(H + 1)$ metamodels in each epoch. The same MM-RGA is used here, but the SF function is modified by replacing $\sum_{j=1}^{J} \langle \widetilde{\underline{g}}_j(\mathbf{x}) \rangle$ term with $\langle \widetilde{\text{ACV}}(\mathbf{x}) \rangle$ in Equation (12). A similar outcome as in Figure 3c occurs here, but the constraints are now handled using one metamodeled $\widetilde{\text{ACV}}(\mathbf{x})$ function. M4-2 does not require an archive to be maintained, as H solutions will be found in one MM-RGA application.

3.5. M5 Framework

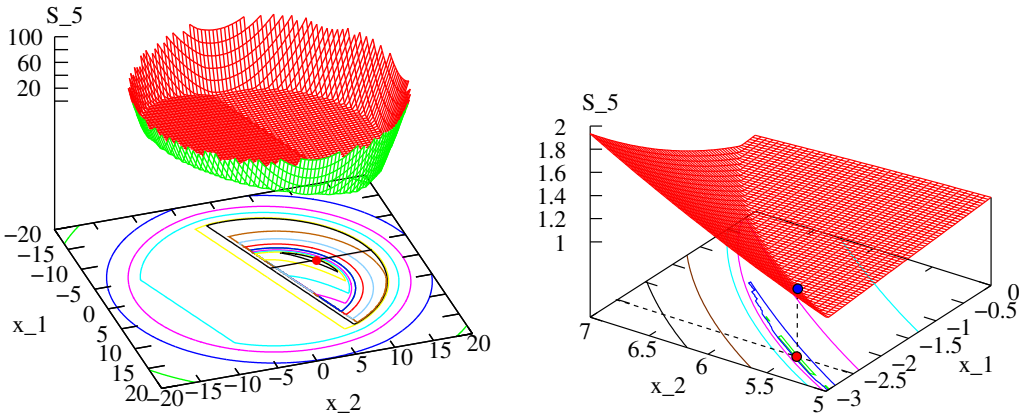
The focus of M5 is to use a generative multiobjective optimization approach in which a single PO solution is found at a time for a $\mathbf{z}$-vector by using a combined *selection* function involving all objective and constraint functions together. The following *selection* function is first created:

$$\mathcal{S}_5(\mathbf{x}, \mathbf{z}) = \begin{cases} \text{ASF}_{34}(\mathbf{x}, \mathbf{z}), & \text{if } \mathbf{x} \text{ is feasible,} \\ \text{ASF}_{34}^{\max}(\mathbf{x}, \mathbf{z}) + \langle \text{ACV}(\mathbf{x}) \rangle, & \text{otherwise.} \end{cases} \tag{15}$$

Here, the parameter $\text{ASF}_{34}^{\max}(\mathbf{x}, \mathbf{z})$ is the worst ASF_{34} function value (described in Equation (9)) of all feasible solutions from the archive. The selection function $\mathcal{S}_5(\mathbf{x}, \mathbf{z})$ is then metamodeled to obtain $\widetilde{\mathcal{S}}_5(\mathbf{x}, \mathbf{z})$, which is then optimized by RGA (described for M1-1) to find one in-fill solution for each $\mathbf{z}$-vector. The unconstrained optimization problem with only variable bounds is given below:

$$\begin{array}{rl} \text{Problem O5:} & \left\{ \begin{array}{ll} \text{Minimize} & \widetilde{S}_5(\mathbf{x}, \mathbf{z}), \\ \text{Subject to} & x_i^{(L)} \leq x_i \leq x_i^{(U)}, \quad i = 1, 2, \ldots, n. \end{array} \right. \\ \text{Solution: } \mathbf{x}^*(\mathbf{z}) & \end{array} \tag{16}$$

Thus, H metamodels of $\mathcal{S}_5(\mathbf{x}, \mathbf{z})$ need to be constructed for M5 in each epoch. Figure 4a shows the S_5 function with $\mathbf{z} = (0.1, 0.9)$ for SRN problem. Although details are not apparent in this figure, Figure 4b, plotted near the optimum, shows optimum more clearly. The entire surface plot is not shown for clarity, but it is interesting to see how a single function differentiates infeasible from feasible region and also makes the optimum of the function as one of the PO solutions.



(a) Optimal solution with $\mathbf{z} = (0.1, 0.9)$ in M5.

(b) The function S_5 surface is blown up near the optimum.

Figure 4. In-fill solution for a specific $\mathbf{z}$-vector to be obtained by framework M5 for SRN problem.

Clearly, the complexity of the resulting $\mathcal{S}_5(\mathbf{x}, \mathbf{z})$ function will demand a large number of archive points for an accurate identification of the PO solution or a large number of epochs to arrive at the PO solution. However, the concept of metamodeling a selection function, which is not one of the original objective or constraint function, to find an in-fill solution of the problem is intriguing and opens up a new avenue for surrogate-assisted multiobjective optimization studies.

3.6. Framework M6

Finally, M6 framework takes the concept of M5 a bit further and constructs a single metamodel in each epoch by combining all M objectives and J constraints together. A multimodal selection function having each optimum corresponding to a distinct PO solution is formed for this purpose:

$$\text{ASF}_6(\mathbf{x}) = \min_{\mathbf{z} \in \mathbf{Z}} \max_{i=1}^{M} \left(f_i(\mathbf{x}) - z_i \right). \tag{17}$$

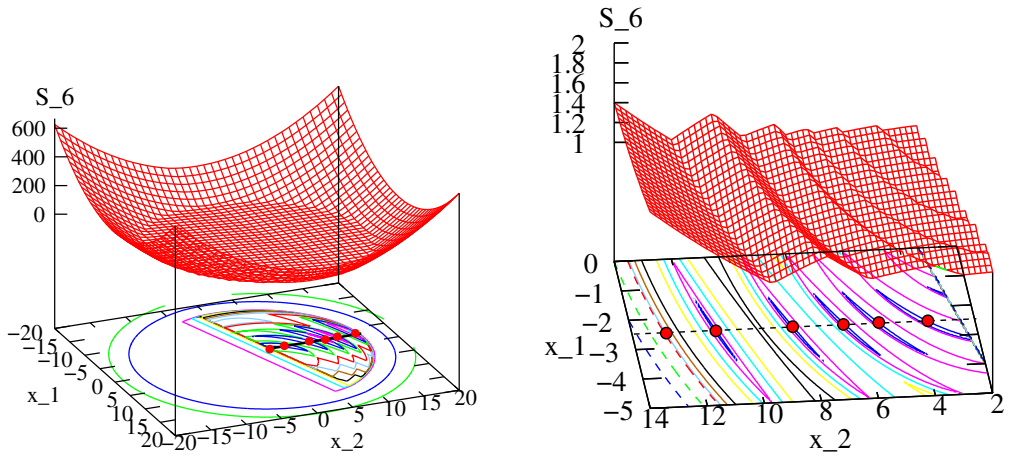
Then, the following selection function is constructed:

$$\mathcal{S}_6(\mathbf{x}) = \left\{ \begin{array}{ll} \text{ASF}_6(\mathbf{x}), & \text{if } \mathbf{x} \text{ is feasible,} \\ \text{ASF}_{6,\max} + \text{CV}(\mathbf{x}), & \text{otherwise,} \end{array} \right. \tag{18}$$

where $\text{ASF}_{6,\max}$ is the maximum ASF_6 value of all feasible archive members. For each archive member $\mathbf{x}$, $\mathcal{S}_6(\mathbf{x})$ is first computed. $\text{CV}(\mathbf{x})$ is same as $\text{ACV}(\mathbf{x})$, except that for a feasible $\mathbf{x}$, CV is set to zero. Then, the following multimodal unconstrained problem (with variable bounds) is constructed to find H in-fill solutions simultaneously:

$$\begin{array}{rl} \text{Problem O6:} & \left\{ \begin{array}{ll} \text{Minimize} & \widetilde{\mathcal{S}}_6(\mathbf{x}), \\ \text{Subject to} & x_i^{(L)} \leq x_i \leq x_i^{(U)}, \quad i = 1, 2, \ldots, n. \end{array} \right. \\ \text{Solutions: } \mathbf{x}^{i,*},\ i = 1, \ldots, H & \end{array} \tag{19}$$

A single metamodel needs to be constructed in each epoch in M6 framework. Due to the complexity involved in the $\mathcal{S}_6$-function, we employ a neural network $\widetilde{\mathcal{S}}_6(\mathbf{x})$ to metamodel this selection function. A niched RGA [7] similar to that described in Section 3.4 is used here to find H in-fill solutions corresponding to each local optimum of the metamodeled $\widetilde{\mathcal{S}}_6(\mathbf{x})$ function. No explicit archive needs to be maintained to store H solutions. Figure 5a shows S_6 function for SRN function on the entire search space. The detail inside the feasible region and near the optimal solutions shown in Figure 5b makes it clear that this function creates six optima on the PO front, corresponding to six $\mathbf{z}$-vectors. Although the function is multimodal, the detail structure from Figure 5a to Figure 5b can be modeled gradually with iterations of a carefully designed optimization algorithm.



(**a**) Multimodal optimal solutions in M6. For illustration, six $\mathbf{z}$-vectors are used in Equation (15).

(**b**) The function S_6 surface is blown up near the optimal region showing six optima.

Figure 5. The function S_6 surface is blown up near the optimal region showing six optima.

3.7. Summary of 10 Frameworks

A summary of metamodeled functions and the optimization algorithms used to optimize them for all 10 frameworks is provided in Table 1. The relative computational cost for each framework can be derived from this table. M3-1 and M3-2 require to construct the maximum number of metamodels (assuming the number of desired PO solutions $H > M$) among all the frameworks, and M6 requires the least, involving only one metamodel in each epoch.

The evolutionary algorithm used to solve each optimization problem is also provided in the table.

Table 1. Summary of metamodeled functions and optimization algorithms needed in each epoch for all 10 frameworks.

Frame-Work	Metamodeling Functions	#Metamodels	Optimization Method	#Opt. Runs
M1-1	$(\underline{f}_1, \cdots, \underline{f}_M)$ $(\underline{g}_1, \cdots, \underline{g}_J)$	$M+J$	RGA	H
M1-2	Same as above	$M+J$	NSGA-II/III	1
M2-1	$(\underline{f}_1, \ldots, \underline{f}_M)$ & ACV	$M+1$	RGA	H
M2-2	Same as above	$M+1$	NSGA-II/III	1
M3-1	ASF_{34} & $(\underline{g}_1, \ldots, \underline{g}_J)$	$H+J$	RGA	H
M3-2	Same as above	$H+J$	MM-RGA	1
M4-1	ASF_{34} & ACV	$H+1$	RGA	H
M4-2	Same as above	$H+1$	MM-RGA	1
M5	$\mathcal{S}_5$	H	RGA	H
M6	$\mathcal{S}_6$	1	N-RGA	1

4. Adaptive Switching Based Metamodeling (ASM) Frameworks

Each metamodeling framework in our proposed taxonomy requires building metamodels for either each objective and constraint or their aggregations. Thus, it is expected that each framework may be most suitable for certain function landscapes that produce a smaller approximation error, but that framework may not be good in other landscapes. During an optimization process, an algorithm usually faces different kinds of landscape complexities from start to finish. Thus, no one framework is expected to perform best during each step of the optimization process. While each framework was applied to different multiobjective optimization problems in another study [6,20] from start to finish, different problems were found to be solved best by different frameworks. To determine the best performing framework for a problem, a simple-minded approach would be to apply each of the 10 frameworks to solve each problem independently using $\text{SE}_{\max}$ high-fidelity evaluations, and then determine the specific framework which performs the best using an EMO metric, such as hypervolume [61] or inverse generational distance (IGD) [62]. This will be computationally expensive, requiring 10 times more than the prescribed $\text{SE}_{\max}$. If each framework is allocated only 1/10-th of $\text{SE}_{\max}$, they may be insufficient to find comparatively good solutions. A better approach would be to use an adaptive switching strategy that chooses the most suitable framework at each epoch.

As mentioned in the previous section, in each epoch, exactly H new in-fill solutions are created irrespective of the metamodeling framework used, thereby consuming H high-fidelity SEs. Clearly, the maximum number of epochs allowable is $E_{\max} = \lceil \frac{SE_{\max}-N_0}{H} \rceil$ with a minor adjustment on the SEs used in the final epoch. At the beginning of each epoch (say, t-th epoch), we have an archive ($\mathcal{A}_t$) of N_t high-fidelity solutions. For the first epoch, these are all N_0 Latin hypercube sampled (LHS) solutions, and in each subsequent epoch, H new in-fill solutions are added to the archive. At the start of t-th epoch, each of the 10 frameworks is used to construct its respective metamodels using all N_t archive members. Then, a 10-fold cross-validation method (described in Section 4.2) is used with a suitable performance metric (described in Section 4.1) to determine the most suitable framework for the next epoch. Thereafter, the best-performing framework is used to find a new set of H in-fill solutions. They are evaluated using high-fidelity evaluations and all 10 frameworks are statistically compared to choose a new best-performing framework for the next epoch. This process is continued until $\text{SE}_{\max}$ evaluations are made. A pseudocode of the proposed ASM approach is provided in Algorithm 1.

Algorithm 1: Adaptive Swithing Framework

Input : Objectives: $[f_1, \ldots, f_m]^T$, Constraints: $[g_1, \ldots, g_J]^T$, frameworks $\mathcal{M}_i$ with parameter Γ_i for $i \in \{1 \ldots, S\}$ where S is number of frameworks, Number of initial samples, allowed high-fidelity solution evaluations, solutions per epoch and cross-validation partitions are N_0, SE_{max}, u and K respectively.

Output: P_T

$t, \text{P}_t, \text{F}_t, \text{G}_t, e \leftarrow 0, \emptyset, \emptyset, \emptyset, N_0$;

$\text{P}_{new} \leftarrow \text{LHS}(\rho)$// Initial sampling

while *True* **do**

  $F_{new} = \{f_i(\text{P}_{new}),\ \forall i \in \{1, \ldots, M\}\}$// high-fidelity objectives eval.

  $G_{new} = \{g_j(\text{P}_{new}),\ \forall j \in \{1, \ldots, J\}\}$// high-fidelity constraints eval.

  $\text{P}_{t+1}, \text{F}_{t+1}, \text{G}_{t+1} \leftarrow (\text{P}_t \cup \text{P}_{new}), (\text{F}_t \cup \text{F}_{new}), (\text{G}_t \cup \text{G}_{new})$// merge pop

  $e \leftarrow e + |\text{P}_{new}|$// number of high-fidelity evaluations

  break if $e \geq \text{SE}_{max}$// termination

  Calculate $\{\text{ASF}(.), \text{ACV}(.), \mathcal{S}_5, \mathcal{S}_6\}$ etc. from P_{t+1}, F_{t+1} & G_{t+1} as per requirements of $\mathcal{M}_i$, $\forall i$;

  Create random K partition (training and test set) Q^k_{t+1} from P_{t+1}, $\forall k \in \{1, \ldots, K\}$;

  for *k=1 to K* **do**

    for *i=1 to S* **do**

      $m_i \leftarrow$ Build corresponding metamodels for framework $\mathcal{M}_i$ using training set of Q^k_{t+1};

      $\text{SEP}(k, i) \leftarrow$ Calculate selection-error probability for m_i with test set of Q^k_{t+1};

  $\mathcal{M}_B \leftarrow$ Identify best frameworks from SEP;

  $\mathcal{M}_b \leftarrow$ Randomly choose a framework from $\mathcal{M}_B$;

  $P_{new} \leftarrow$ Optimize framework $\mathcal{M}_b(m_b, \Gamma_b)$;

  if $|P_{t+1}| + |P_{new}| > SE_{max}$ **then**

    $P_{new} \leftarrow$ Randomly pick $\text{SE}_{max} - |P_{t+1}|$ solutions from P_{new};

  $t \leftarrow t + 1$;

  // end of epoch

return $\text{P}_T \leftarrow$ filter best solutions from P_{t+1}

4.1. Performance Metric for Framework Selection

To compare the performances among multiple surrogate models, mean squared error (MSE) has been widely used in literature [30]. For optimization algorithms, the regression methods that use MSE are known to be susceptible to outliers. For multiple objectives, different objectives and constraints may have different scaling. Our pilot study shows that even with the normalization of the objectives and constraints, the MSE metric does not always correctly evaluate the metamodels. Here, we introduce a *selection error probability* (SEP) metric which is more appropriate for an optimization task than MSE metric or even other measures, such as, the Kendal rank correlation coefficient [63] metric. The usual metrics may be better for a regression task, but for an optimization task, the proposed SEP makes a more direct evaluation of pair-wise comparisons of solutions.

SEP is defined as the probability of making an error in correctly predicting the better of two solutions compared against each other using the constructed metamodels. Consider Figure 6, which illustrates an minimization task and comparison of three different population members pair-wise. The true function values are shown in solid blue, while the predicted function values are shown in dashed blue. When points x_1 and x_2 are compared based on predicted function, the prediction is correct, since $f((x_1) < f(x_2)$ and also $\tilde{f}(x) < \tilde{f}(x_2)$. However, when points x_1 and x_3 are compared against each other, the prediction is wrong. Out of the three pairwise comparisons, two predictions are correct and one is wrong, thereby making a selection error probability of $1/3$ for this case. We argue that in an optimization procedure, it is the SEP which provides a better selection error than the actual function values, as the relative function values are important than the exact function values.

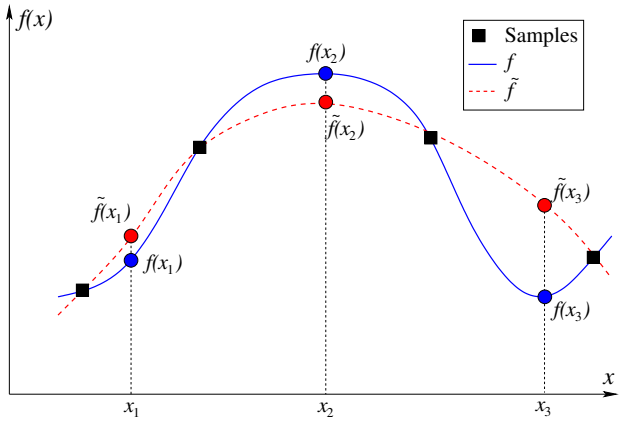


Figure 6. Selection Error Probability (SEP) concept is illustrated.

Mathematically, the SEP metric can be defined for n points as follows. For each of $N = \binom{n}{2}$ pairs of points (p and q), we evaluate the selection error function ($E(p,q)$), which is one, if there is a mismatch between predicted winner and actual winner of p and q; zero, otherwise. Then, SEP is calculated as follows:

$$\text{SEP} = \frac{1}{N}\sum_{p=1}^{n-1}\sum_{q=p+1}^{n} E(p,q). \tag{20}$$

The definition of a "winner" can be easily extended to multiobjective and constrained multiobjective optimization by considering the domination [64] and constraint-domination [54] status of two points p and q.

4.2. Selecting a Framework for an Epoch

Frameworks having least SEP value are considered to be the best for performing the next epoch. We have performed 10-fold cross-validation in order to identify the best frameworks. After each epoch, H new in-fill points are evaluated using high-fidelity evaluations and added to the archive. In each fold of cross-validation, 90% solutions are used for constructing metamodels with respect to the competing frameworks. Then the corresponding frameworks are used to compare every pair (p and q) of the remaining 10% of archive points using the SEP metric. We apply constrained domination checks to identify the relationship between these two solutions. We then compare this relationship with the true relationship given by their high-fidelity values with the same constrained domination check. We calculate the selection error function ($E(p,q)$) for each pair of test archive solutions. The above process is repeated 10 times by using different blocks of 90% points to obtain 10 different SEP values for each framework. This cross-validation procedure does not require any new solution evaluations, as the whole computations are performed based on the already-evaluated archive points and their predicted values from each framework. Thereafter, the best framework is identified based on the median SEP value of frameworks.

Finally, the Wilcoxon rank-sum test is performed between the best framework and all other frameworks. All frameworks within a statistical insignificance (having $p > 0.05$) are identified to obtain the best-performing set $\mathcal{M}_B$. Then a randomly chosen framework ($\mathcal{M}_b$) is selected from $\mathcal{M}_B$ for the next epoch. Since each of these frameworks performs similarly in a sense of median performance, the choice of a random framework makes the ASM approach diverse with the probability of using different metamodeling landscapes in successive epochs. This procedure, in practice, prohibits the overall approach from getting stuck in similar metamodeling frameworks for long, even it is one of the best performing frameworks.

4.3. Trust-Region Based Real-Coded Genetic Algorithms

Before we present the results, we need to discuss one other algorithmic aspect, which is important. Since the metamodels are not error-free, predictions of solutions close to high-fidelity solutions are usually more accurate than predictions far from them. Therefore, we use a trust-region method [65] in which predictions are restricted within a radius R_{trust} from each high-fidelity solution in the variable space. Trust region method is used in nonevolutionary metamodeling studies [35,36]. Another parameter R_{prox} is also introduced which defines the minimum distance with which any new solution should be located from an archive member to provide a diverse set of in-fill solutions. We simulate a feasible search region R_{search} around every high-fidelity solution: $R_{prox} \leq R_{search} \leq R_{trust}$. Using the concepts of trust-region method from the literature [66], we reduce the two radii at every epoch by constant factors: $R^{\text{new}}_{trust} = 0.75R^{\text{old}}_{trust}$ and $R^{\text{new}}_{prox} = 0.1R^{\text{new}}_{trust}$. A reduction of two radii helps in achieving more trust on closer to high-fidelity solutions with iterations. These factors are found to perform well on a number of trial-and-error studies prior to obtaining the results presented in the next section.

The optimization methods for metamodels are modified as follows. At generation t, parent population P_t is applied by a standard binary constrained tournament selection on two competing population members using the metamodeled objectives, constraints, or selection criteria described before to choose the winner. Standard recombination and mutation operators (without any care for trust region concept) are used to create an offspring population, which is then combined with the parent population P_t and then better half is chosen for the next generation as parent population P_{t+1} using the trust region concept. We first count the number of solutions in the combined population within the two trust regions. If the number is smaller than or equal to N, then they are copied to P_{t+1} and remaining slots are filled with solutions which are closest to the high-fidelity solutions in the variable space. On the other hand, if the number is larger than N, the same binary constrained tournament selection method is applied to pick N solutions from them and copied to P_{t+1}.

5. Results and Discussion

We present the results of the ASM approach on 18 different test and engineering problems. The problems include two to five-objective, constrained, and unconstrained problems. In order to get robust performance, we have included all 10 frameworks as options for switching in our ASM approach. The performance of ASM approach is compared with each framework alone. We then compare ASM's performance with three recently suggested multiobjective metamodeling methods: MOEA/D-EGO [14], K-RVEA [23], and CSEA [33].

5.1. Parameter Settings

For two-objective problems, we use NSGA-II [39] for M1-2 and M2-2 frameworks. For problems with higher number of objectives, we use NSGA-III [60] procedure. Note that, other multiobjective evolutionary algorithms (e.g., MOEA/D [14] or RVEA [23]) can also be used. A population of size ($N = 100$) is used when the number of reference lines (H) is less than 100. Otherwise, the population size is set identical to H. Initial archive size is set according to Table 2. Other parameter settings are as follows: number of generations $\tau = 300$, SBX crossover probability $p_c = 0.95$, polynomial mutation probability $p_m = 1/n$ (where n is the number of variables), distribution indices for SBX and mutation operators are $\eta_c = 20$ and $\eta_m = 20$, respectively. Initial value of R_{trust} is set to be $\sqrt{n}$ for the normalized problems having variable domain $[0, 1]^n$. The number of reference points, SE_{max}, resulting epochs for each problem are presented in Table 2.

Table 2. Parameter values for 18 problems.

Problem	n	M	J	N_0	SE_{max}	H	#Epochs
ZDT1	10	2	0	100	500	21	20
ZDT2	10	2	0	100	500	21	20
ZDT3	10	2	0	100	500	21	20
ZDT4	5	2	0	100	1000	21	43
ZDT6	10	2	0	100	500	21	20
OSY	6	2	6	200	800	21	29
TNK	2	2	2	200	800	21	29
SRN	2	2	2	200	800	21	29
BNH	2	2	2	200	800	21	29
WB	4	2	4	300	1000	21	39
DTLZ2	7	3	0	500	1000	91	6
C2DTLZ2	7	3	1	700	1500	91	9
CAR	7	3	10	700	2000	91	15
DTLZ5	7	3	0	500	1000	91	6
DTLZ4	7	3	0	700	2000	91	15
DTLZ7	7	3	0	500	1000	91	6
DTLZ2-5	7	5	0	700	2500	210	9
C2DTLZ2-5	7	5	1	700	2500	210	9

5.2. Two-Objective Unconstrained Problems

First, we apply our proposed methodologies to two-objective unconstrained problems: ZDT1, ZDT2, ZDT3, ZDT4 and ZDT6. Table 3 presents the median IGD values of 11 runs for each framework applied standalone from start to end. In the absence of any constraint or having a single constraint, M1-1 and M2-1 are identical frameworks; so are M1-2 and M2-2, M3-1 and M4-1, M3-2, and M4-2. This is why we keep a blank under M2-1, M2-2, M4-1, M4-2 entries for unconstrained and single-constraint problems in the table. The best performing method is first identified based on the median IGD values and is marked in bold. A p-value from an Wilcoxon rank sum test of each other method is then computed for 11 runs with the 11 runs of the best-performing method. If any algorithm produces a p-value greater than 0.05, it indicates that the algorithm has produced a statistically similar performance to the best-performing method and its median IGD value is then marked in italics. It is clear from the table that the ASM approach (right-most column), being mostly in bold, performs better or equivalent to all frameworks for all five ZDT problems, whereas M1-1 performs the best in the first four problems. M1-2 and M3-1 performs well in three test problems, whereas M6 performs the best in ZDT6 problem. Obtained nondominated solutions of two-objective constrained and unconstrained problems of the median run are presented in Figure 7. We also show performance of other comparing algorithms: MOEA/D-EGO [14], K-RVEA [23], and CSEA [33] in the figure.

It is apparent that ASM approach is able to find a better distributed and converged set of points than other methods for an identical number of SEs.

The epoch-wise proportion of usage of each framework over 11 runs of the ASM approach is shown in Figure 8 for all five ZDT problems. For ZDT1, standalone M1-1, M2-1, M3-1, and M4-1 perform in a statistically similar manner as shown in Table 3, but the ASM approach mostly restricts its epoch-wise choice on M1-1, M1-2, M2-1, and M2-2 and produces a similar performance in most epochs. Since multiple frameworks can appear with a similar performance in an epoch, the proportions (shown in Figure 8) need not sum up to one at each epoch. For ZDT2, only M1-1 and M1-2 perform well as a standalone framework (Table 3), and the ASM approach is able to pick these two frameworks to produce the best performing result. Notice that since ZDT1 and ZDT2 do not have any constraint, M1-1 and M2-1 are identical frameworks and M1-2 and M2-2 are identical frameworks. Except in ZDT6, M1-1, M1-2, M1-2, and M2-2, for which objectives are independently metamodeled, turn to be dominating frameworks. However, for ZDT6, M3-2, M4-2, and M6 show their dominance. In ZDT4, almost all the frameworks are found to be switching between them early on but settles with M1 and M2 frameworks at the latter

part of the optimization runs. Switching among different frameworks performs well on all five problems.

The switching patterns of frameworks for the median performing run for ZDT1, ZDT4, and ZDT6 are shown in Figure 9. Although multiple frameworks may exist at the end of each epoch, the figure shows the specific framework which was chosen for this specific run. For ZDT2, the ASM approach juggles mostly between M1 and M2 variants and produce the best performing result, even better than M1 and M2 alone. In ZDT4, the ASM approach alternates between eight frameworks in the beginning and settles with four of them (M3 and M4 variants) in the middle and then uses M3 variants at the end to produce statistically equivalent result to M1-1 alone. Interestingly, while as a standalone framework from start to end, M1-1 performs the best performance, the ASM approach does not use M1-1 in any of the epochs. The switching of different frameworks from epoch to epoch is clear from these plots.

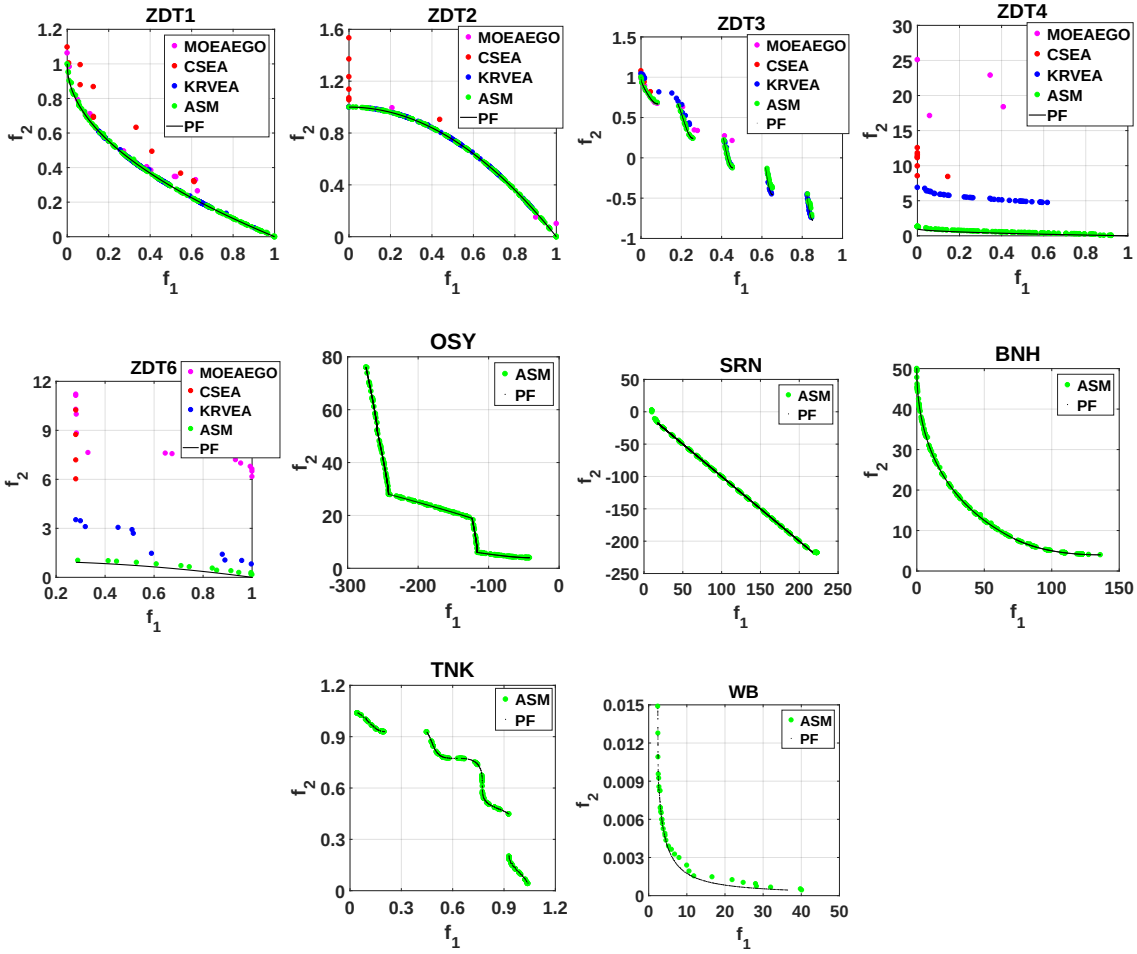


Figure 7. Non-dominated solutions of the final archive for the median run of ASM approach for two-objective ZDT and constrained problems. In all cases, a well-diversified set of near PO solutions is obtained with a limited solution evaluations.

5.3. Two-Objective Constrained Problems

Next, we apply ASM approach and all the frameworks separately to standard two-objective constrained problems: BNH, SRN, TNK, OSY, and the welded beam problem (WB) [54]. The ASM approach performs the best on three of the five problems, followed by M1-1 which performed best in two problems; however, both these methods perform the best statistically on all five problems. Other individual frameworks do not perform so well. Figure 8 shows the epoch-wise utilization of different frameworks for TNK and WB

in 11 runs. The plots for TNK shows that ASM almost always chooses M1-1 or M1-2 as the best-performing frameworks as supported by IGD values in Table 3.

However, on WB problem, ASM approach selects M1-1, M5, and M6 in most of the epochs, despite poor performance of the latter two when applied in a stand-alone manner from start to end.

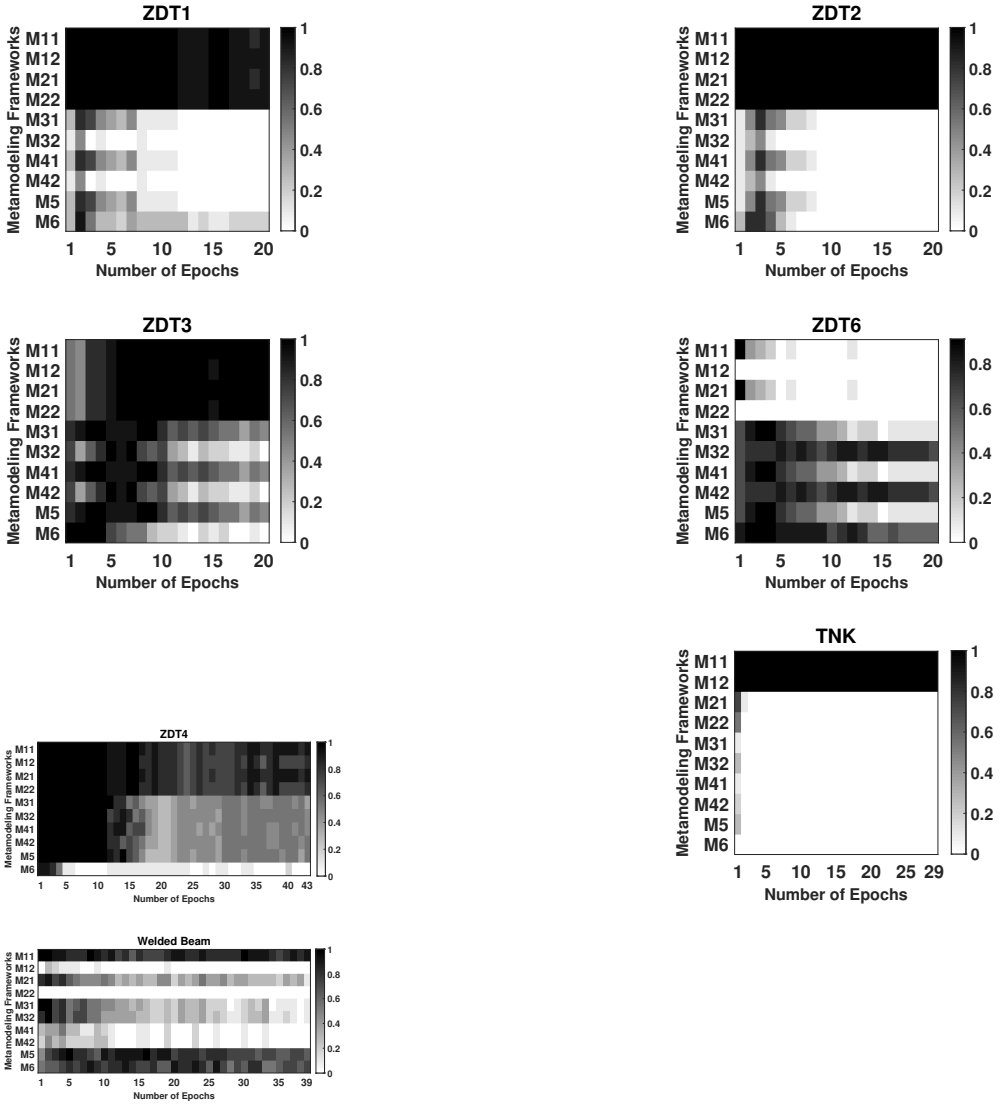


Figure 8. Epoch-wise proportion of appearance of 10 frameworks within M_B in 11 runs of the ASM approach for ZDT problems, TNK, and welded beam design problems indicates the use of multiple frameworks during optimization. Some problems uses some specific frameworks more frequently.

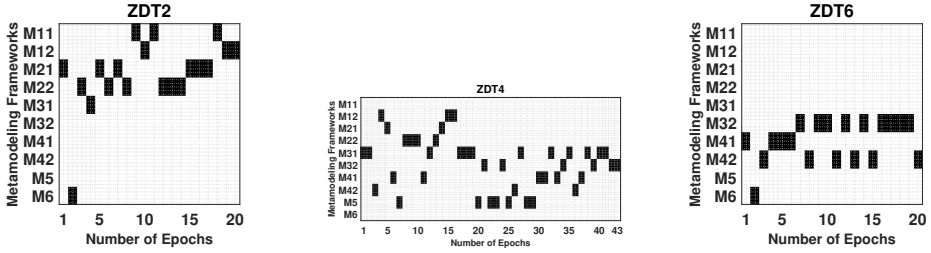


Figure 9. Switching among frameworks for the median IGD run of the ASM approach for ZDT2, ZDT4, and ZDT6 indicates that many frameworks are used during the optimization process.

Table 3. IGD values obtained from all the individual frameworks and proposed switching algorithm on different test problems are presented. The best performing framework and other statistically similar frameworks are marked in bold with their *p*-values in the second row. For problems without any constraint, the framework M*i*-1 is identical to M*i*-2, hence a "-" is provided for the latter. For unconstrained problems, M5 and M6 are also identical.

Problem	M1-1	M2-1	M1-2	M2-2	M3-1	M4-1	M3-2	M4-2	M5	M6	ASM
ZDT1	**0.00090** -	- -	***0.00555*** $p = 0.4701$	- -	***0.00447*** $p = 0.4702$	- -	***0.00537*** $p = 0.7928$	- -	- -	0.01337 $p = 8.1\times10^{-5}$	***0.00130*** $p = 0.091$
ZDT2	***0.00065*** $p = 0.2372$	- -	***0.00062*** $p = 0.2372$	- -	0.00568 $p = 8.1\times10^{-5}$	- -	0.00910 $p = 8.1\times10^{-5}$	- -	- -	0.72366 $p = 8.1\times10^{-5}$	**0.00055** -
ZDT3	***0.00531*** $p = 0.325$	- -	**0.00212** -	- -	0.17123 $p = 8.1\times10^{-5}$	- -	0.19050 $p = 8.1\times10^{-5}$	- -	- -	0.08315 $p = 8.1\times10^{-5}$	***0.00391*** $p = 0.369$
ZDT4	**0.28900** -	- -	5.43450 $p = 8.1\times10^{-5}$	- -	***0.29300*** $p = 0.4307$	- -	0.43450 $p = 0.0126$	- -	- -	6.15510 $p = 8.1\times10^{-5}$	***0.39992*** $p = 0.1310$
ZDT6	***0.37058*** $p = 0.2934$	- -	0.48360 $p = 8.1\times10^{-5}$	- -	***0.24192*** $p = 0.8438$	- -	0.47159 $p = 0.0013$	- -	- -	**0.21327** -	***0.24440*** $p = 0.3933$
OSY	***0.15323*** $p = 0.2301$	24.57940 $p = 8.1\times10^{-5}$	0.18806 $p = 8.1\times10^{-5}$	22.99990 $p = 8.1\times10^{-5}$	6.26550 $p = 8.1\times10^{-5}$	18.49200 $p = 8.1\times10^{-5}$	4.77670 $p = 8.1\times10^{-5}$	18.33760 $p = 8.1\times10^{-5}$	45.18110 $p = 8.1\times10^{-5}$	57.15870 $p=8.1\times10^{-5}$	**0.12110** -
TNK	**0.00073** -	0.04383 $p = 8.1\times10^{-5}$	***0.00082*** $p = 0.206$	0.02849 $p = 8.1\times10^{-5}$	0.01180 $p = 8.1\times10^{-5}$	0.03332 $p = 8.1\times10^{-5}$	0.01121 $p = 8.1\times10^{-5}$	0.03743 $p = 8.1\times10^{-5}$	0.03077 $p = 8.1\times10^{-5}$	0.03990 $p = 8.1\times10^{-5}$	***0.00080*** $p = 0.494$
SRN	**0.13191** -	4.17160 $p = 8.1\times10^{-5}$	1.00930 $p = 8.1\times10^{-5}$	0.92614 $p = 8.1\times10^{-5}$	1.06120 $p = 8.1\times10^{-5}$	1.20480 $p = 8.1\times10^{-5}$	1.51360 $p = 8.1\times10^{-5}$	1.48870 $p = 8.1\times10^{-5}$	1.28450 $p = 8.1\times10^{-5}$	2.41710 $p = 8.1\times10^{-5}$	***0.13406*** $p = 0.1891$
BNH	***0.07885*** $p = 0.0865$	0.74425 $p = 8.1\times10^{-5}$	***0.04630*** $p = 0.5114$	***0.04457*** $p = 0.5994$	0.23728 $p = 8.1\times10^{-5}$	0.23923 $p = 8.1\times10^{-5}$	0.32874 $p = 8.1\times10^{-5}$	0.36600 $p = 8.1\times10^{-5}$	0.23699 $p = 8.1\times10^{-5}$	0.71300 $p = 8.1\times10^{-5}$	**0.04176** -
WB	***0.13794*** $p = 0.2933$	0.55529 $p = 8.1\times10^{-5}$	0.23159 $p = 0.0126$	0.84746 $p = 8.1\times10^{-5}$	***0.16909*** $p = 0.1007$	0.88586 $p = 8.1\times10^{-5}$	1.39250 $p = 8.1\times10^{-5}$	3.40770 $p = 8.1\times10^{-5}$	0.96166 $p = 8.1\times10^{-5}$	1.41110 $p = 8.1\times10^{-5}$	**0.08960** -

Table 3. *Cont.*

Problem	M1-1	M2-1	M1-2	M2-2	M3-1	M4-1	M3-2	M4-2	M5	M6	ASM
DTLZ2	0.07870 $p = 8.1\times10^{-5}$	- -	**0.03340** -	- -	0.05377 $p = 8.1\times10^{-5}$	- -	0.05040 $p = 8.1\times10^{-5}$	- -	- -	0.07736 $p = 8.1\times10^{-5}$	***0.03701*** $p = 0.562$
C2DTLZ2	0.05130 $p = 8.1\times10^{-5}$	- -	***0.03355*** $p = 0.115$	- -	0.03493 $p = 0.008$	- -	***0.03190*** $p = 0.148$	- -	0.12403 $p = 8.1\times10^{-5}$	0.04410 $p = 8.1\times10^{-5}$	**0.03062** -
CAR	0.43510 $p = 8.1\times10^{-5}$	0.43145 $p = 8.1\times10^{-5}$	0.50119 $p = 8.1\times10^{-5}$	**0.29817** -	0.39809 $p = 8.1\times10^{-5}$	0.42223 $p = 8.1\times10^{-5}$	0.40494 $p = 8.1\times10^{-5}$	0.44251 $p = 8.1\times10^{-5}$	0.50061 $p = 8.1\times10^{-5}$	0.55569 $p = 8.1\times10^{-5}$	0.40110 $p = 8.1\times10^{-5}$
DTLZ5	0.01960 $p = 8.1\times10^{-5}$	- -	**0.00948** -	- -	0.01352 $p = 8.1\times10^{-5}$	- -	0.01537 $p = 8.1\times10^{-5}$	- -	- -	0.05421 $p = 8.1\times10^{-5}$	***0.01252*** $p = 0.0605$
DTLZ4	**0.05840** -	- -	***0.09024*** $p = 0.1203$	- -	0.20668 $p = 8.1\times10^{-5}$	- -	0.12570 $p = 8.1\times10^{-5}$	- -	- -	***0.08731*** $p = 0.3933$	***0.07934*** $p = 0.425$
DTLZ7	0.11808 $p = 0.0187$	- -	***0.07664*** $p = 0.2122$	- -	0.87172 $p = 8.1\times10^{-5}$	- -	1.26300 $p = 8.1\times10^{-5}$	- -	- -	0.82989 $p = 8.1\times10^{-5}$	**0.06529** -
DTLZ2-5	0.21450 $p = 8.1\times10^{-5}$	- -	**0.03981** -	- -	0.14401 $p = 8.1\times10^{-5}$	- -	0.14403 $p = 8.1\times10^{-5}$	- -	- -	0.11028 $p = 8.1\times10^{-5}$	***0.04918*** $p = 0.595$
C2DTLZ2-5	0.17341 $p = 8.1\times10^{-5}$	- -	***0.03676*** $p = 0.8541$	- -	0.15388 $p = 8.1\times10^{-5}$	- -	0.11669 $p = 8.1\times10^{-5}$	- -	0.29291 $p = 8.1\times10^{-5}$	0.20842 $p = 8.1\times10^{-5}$	**0.03441** -

5.4. Three and More Objective Constrained and Unconstrained Problems

Next, we apply all ten frameworks and ASM approach to three-objective optimization problems (DTLZ2, DTLZ4, DTLZ5, and DTLZ7) and also to two three-objective constrained problem (C2DTLZ2 and the car side impact problem CAR [60]). Table 3 shows that while M2-2 works uniquely the best on CAR, M1-2 and M3-2 on C2-DTLZ2, and M1-1, M1-2, and M6 on DTLZ4, the performance of ASM approach is better or equivalent compared to all 10 problems.

The epoch-wise proportion of utilization of 10 frameworks in 11 runs are shown in Figure 10 for three and five-objective problems. It can be clearly seen that M3-1 to M6 frameworks are not usually chosen by the ASM approach on most of these problems, except for complex problems, such as DTLZ4. Switching has been confined between M1-1 to M2-2 for most problems, except in DTLZ4, in which all generative frameworks are found to be useful in certain stages during the optimization process. DTLZ7 works better with simultaneous frameworks M1-2 and M2-2.

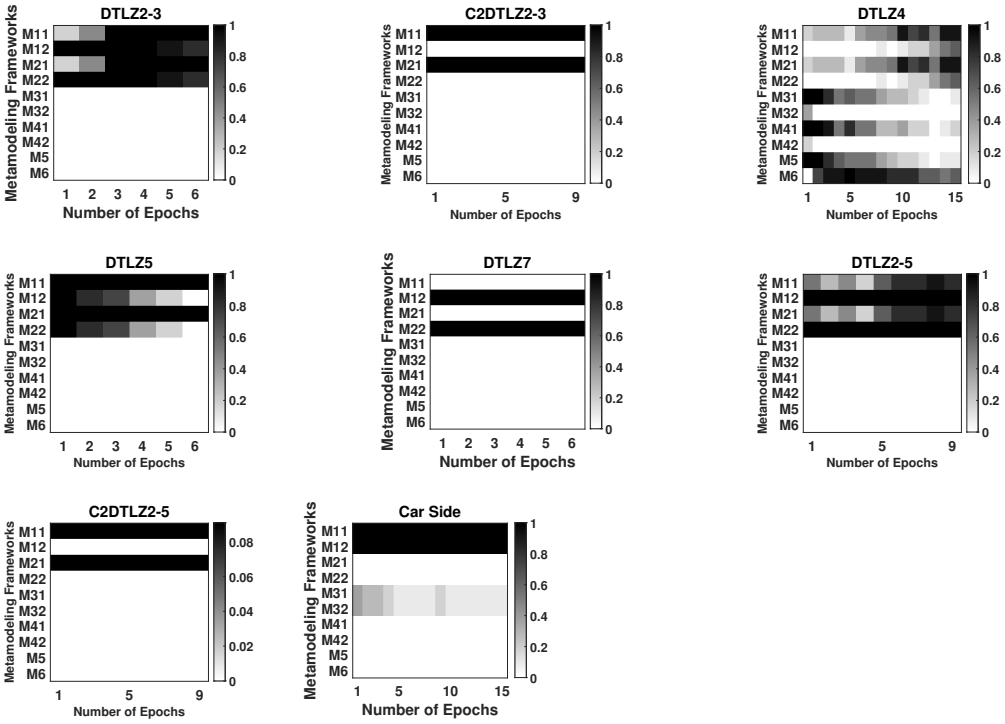


Figure 10. Epoch-wise proportion of usage of 10 frameworks in 11 runs of the ASM approach for three and five-objective problems.

On two five-objective unconstrained DTLZ2 and constrained C2-DTLZ2 problems, M1-2 alone and ASM approach perform the best with statistically significant difference with other frameworks. Constrained C2DTLZ2 problems use similar a switching pattern for three and five-objective version of the problem.

Table 4 calculates the rank of each of the 10 frameworks for solving 18 problems. The table shows that the ASM approach performs the best overall, followed by M1-2, M2-2, and M3-1 respectively. It indicates that overall, metamodeling of objectives independently is a better approach for these problems. M6, although being the most efficient in the number of metamodels, performs the worst.

Table 4. Average rank of 10 frameworks and the ASM approach on 18 problems based on Wilcoxon rank-sum test.

M1-1	M2-1	M1-2	M2-2	M3-1	M4-1	M3-2	M4-2	M5	M6	ASM
3.66	6.16	2.88	3.00	4.55	5.44	6.22	6.94	6.33	8.55	**1.11**

5.5. Comparison with Existing Methods

Next, we examine the performance of our adaptive switching metamodeling (ASM) strategy by comparing them with a few recent algorithms, namely, MOEA/D-EGO [14], K-RVEA [23], and CSEA [33]. Algorithms are implemented in PlatEMO [67]. Since these three competing algorithms can only be applied to unconstrained problems, only ZDT and DTLZ problems are considered here. We plan to compare our constrained approach with existing constraint handling methods [32]. Identical parameters settings as those used with the ASM approach are used for the three competing algorithms. Table 5 presents the mean IGD value of each algorithm. The Wilcoxon rank-sum test results are also shown. It is clearly evident that ASM approach outperforms three competing methods, of which K-RVEA performs well only on two of the nine problems.

Table 5. Median IGD on unconstrained problems using ASM approach, and MOEA/D-EGO, K-RVEA, and CSEA algorithms. DNC is denoted as "Did not converge" within given time.

Problem	MOEA/D-EGO	K-RVEA	CSEA	ASM
ZDT1	0.05611 $p = 8.1\times10^{-5}$	0.07964 $p = 8.1\times10^{-5}$	0.95330 $p = 8.1\times10^{-5}$	**0.00130** $p = 0.0910$
ZDT2	0.04922 $p = 8.1\times10^{-5}$	0.03395 $p = 8.1\times10^{-5}$	1.01060 $p = 8.1\times10^{-5}$	**0.00055** -
ZDT3	0.30380 $p = 8.1\times10^{-5}$	0.02481 $p = 8.1\times10^{-5}$	0.94840 $p = 8.1\times10^{-5}$	**0.00391** -
ZDT4	73.25920 $p = 8.1\times10^{-5}$	4.33221 $p = 8.1\times10^{-5}$	12.71600 $p = 8.1\times10^{-5}$	**0.39992** -
ZDT6	0.51472 $p = 8.1\times10^{-5}$	0.65462 $p = 8.1\times10^{-5}$	5.42620 $p = 8.1\times10^{-5}$	**0.24440** $p = 0.0612$
DTLZ2	0.33170 $p = 8.1\times10^{-5}$	0.0548 $p = 8.1\times10^{-5}$	0.11420 $p = 8.1\times10^{-5}$	**0.03701** $p = 0.157$
DTLZ4	0.64533 $p = 8.1\times10^{-5}$	**0.0449** -	0.08110 $p = 0.0022$	0.07934 $p = 0.0380$
DTLZ5	0.26203 $p = 8.1\times10^{-5}$	0.0164 $p = 8.1\times10^{-5}$	0.03081 $p = 8.1\times10^{-5}$	**0.01252** $p = 0.211$
DTLZ7	5.33220 $p = 8.1\times10^{-5}$	**0.0531** -	0.70520 $p = 8.1\times10^{-5}$	**0.06529** $p = 0.1930$
DTLZ2-5	0.31221 $p = 8.1\times10^{-5}$	0.23031 $p = 8.1\times10^{-5}$	DNC DNC	**0.04918** -

6. Conclusions

In this paper, we have provided a brief review of existing metamodeling methods for multiobjective optimization, since there has been a surge in such studies in the recent past. Since this calls for modeling multiple objectives and constraints in a progressive manner, a recently proposed taxonomy of 10 frameworks involving metamodeling of independent or aggregate functions of objectives and constraints have been argued to cover a wide variety such methods. Each framework has been presented in detail, comparing and contrasting them in terms of the number of metamodeling functions to be constructed, the number of internal optimization problems to be solved, and the type of optimization methods to be employed, etc. We have argued that each metamodeling framework may be ideal at different stages during an optimization run on an arbitrary problem, hence, an ensemble use of all 10 frameworks becomes a natural choice. To propose an efficient

multiobjective metamodeling algorithm, we have proposed an adaptive switching based metamodeling (ASM) methodology which automatically chooses the most appropriate framework epoch-wise during the course of an optimization run. In order to choose the best framework in every epoch, we perform statistical tests based on a newly proposed acceptance criterion—selection error probability (SEP), which counts the correct pairwise relationships of objectives between two test solutions in a k-fold cross-validation test, instead of calculating the usual mean-squared error of metamodeled objective values from true values. We have observed that SEP is less sensitive to outliers and is much better suited for multiobjective constrained optimization. In each epoch, the ASM approach switches to an appropriate framework which then creates a prespecified number of in-fill points by using either an evolutionary single or multiobjective algorithm or by using a multimodal or a niche-based real-parameter genetic algorithm. On 18 test and engineering problems having two to five objectives and multiple constraints, the ASM approach has been found to perform much better compared to each framework alone and also to three other existing metamodeling multiobjective algorithms.

It has been observed that in most problems a switching between different M1 and M2 frameworks, in which objectives are independently metamodeled, has performed the best. Metamodeling of constraints in an aggregate manner or independently is not an important matter. However, for more complex problems, such as ZDT3, ZDT6, ZDT4, DTLZ4, and engineering design problems, all 10 frameworks, including M5 and M6, have been involved at different stages of optimization. Interestingly, certain problems have preferred to pick generative frameworks (Mi-1 and M5) only, while some others have preferred simultaneous frameworks (Mi-2 and M6). Clearly, further investigation is needed to decipher a detail problem-wise pattern of selecting frameworks, but this first study on statistics-based adaptive switching has clearly shown its advantage over each framework applied alone.

While in this paper, Kriging metamodeling method has been used for all frameworks, this study can be extended to choose the best metamodeling method from an ensemble of RBF, SVR, or other response surface methods to make the overall approach more computationally efficient. In many practical problems, some functions may be relatively less time-consuming, thereby creating a *heterogeneous* metamodeling scenario [16,68,69]. A simple extension of this study would be to formulate a heterogeneous *MP* (for example, M1-1's objective function for a two-objective problem involving a larger evaluation time for f_1 can be chosen as $\left(\underline{\widetilde{f}}_1(\mathbf{x}), \underline{f}_2(\mathbf{x})\right)$, in which the objective f_2 has not been metamodeled at all). However, more involved algorithms can be tried for to handle such pragmatic scenarios. Another practical aspect comes from the fact that a cluster of objectives and constraints can come at the end of a single expensive evaluation procedure (such as, compliance objective and stress constraint comes after an expensive finite element analysis on a mechanical component design problem), whereas other functions come from a different time-scale evaluation procedure. The resulting definition of an epoch and the overall metamodeling approach need to be reconsidered to make the overall approach efficient. Other tricks, such as, the use of a low-fidelity evaluation scheme for expensive objective and constraints early on during the optimization process using a multifidelity scheme and the use of domain-informed heuristics to initialize population and repair offspring solutions must also be considered while developing efficient metamodeling approaches.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Cassioli, A.; Schoen, F. Global optimization of expensive black box problems with a known lower bound. *J. Glob. Optim.* **2013**, *57*, 177–190. [CrossRef]
2. Jin, Y. Surrogate-assisted evolutionary computation: Recent advances and future challenge. *Swarm Evol. Comput.* **2011**, *1*, 61–70. [CrossRef]

3. Ponweiser, W.; Wagner, T.; Biermann, D.; Vincze, M. Multiobjective Optimization on a Limited Budget of Evaluations Using Model-Assisted $\mathcal{S}$-Metric Selection. In *Parallel Problem Solving from Nature–PPSN X*; Springer: Berlin/Heidelberg, Germany, 2008; pp. 784–794.
4. Jones, D.R. A taxonomy of global optimization methods based on response surfaces. *J. Glob. Optim.* **2001**, *21*, 345–383. [CrossRef]
5. Hussein, R.; Deb, K. A Generative Kriging Surrogate Model for Constrained and Unconstrained Multi-objective Optimization. In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '16), Denver, CO, USA, 20–24 July 2016; ACM Press: New York, NY, USA, 2016.
6. Deb, K.; Hussein, R.; Roy, P.; Toscano, G. Classifying Metamodeling Methods for Evolutionary Multi-objective Optimization: First Results. In *Evolutionary Multi-Criterion Optimization EMO*; Springer: Berlin/Heidelberg, Germany, 2017.
7. Roy, P.; Hussein, R.; Deb, K. Metamodeling for multimodal selection functions in evolutionary multi-objective optimization. In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '17), Berlin, Germany, 15–19 July 2017; ACM Press: New York, NY, USA, 2017.
8. Bhattacharjee, K.S.; Singh, H.K.; Ray, T. Multi-objective optimization with multiple spatially distributed surrogates. *J. Mech. Des.* **2016**, *138*, 091401. [CrossRef]
9. Bhattacharjee, K.S.; Singh, H.K.; Ray, T.; Branke, J. Multiple Surrogate Assisted Multiobjective Optimization Using Improved Pre-Selection. In Proceedings of the 2016 IEEE Congress on Evolutionary Computation (CEC-2016), Vancouver, BC, Canada, 24–29 July 2016.
10. Emmerich, M.T.M.; Giannakoglou, K.C.; Naujoks, B. Single- and multiobjective evolutionary optimization assisted by Gaussian random field metamodels. *IEEE Trans. Evol. Comput.* **2006**, *10*, 421–439. [CrossRef]
11. Byrd, R.H.; Nocedal, J.; Waltz, R.A. Knitro: An Integrated Package for Nonlinear Optimization. In *Large-Scale Nonlinear Optimization*; Springer US: Boston, MA, USA, 2006.
12. Jin, Y.; Oh, S.; Jeon, M. Incremental approximation of nonlinear constraint functions for evolutionary constrained optimization. In Proceedings of the 2010 IEEE Congress on Evolutionary Computation (CEC-2010), Barcelona, Spain, 18–23 July 2010; pp. 1–8.
13. Datta, R.; Regis, R.G. A surrogate-assisted evolution strategy for constrained multi-objective optimization. *Expert Syst. Appl.* **2016**, *57*, 270–284. [CrossRef]
14. Zhang, Q.; Liu, W.; Tsang, E.; Virginas, B. Expensive Multiobjective Optimization by MOEA/D With Gaussian Process Model. *IEEE Trans. Evol. Comput.* **2010**, *14*, 456–474. [CrossRef]
15. Knowles, J. ParEGO: A Hybrid Algorithm with On-line Landscape Approximation for Expensive Multiobjective Optimization Problems. *IEEE Trans. Evol. Comput.* **2006**, *10*, 50–66. [CrossRef]
16. Allmendinger, R.; Emmerich, M.T.; Hakanen, J.; Jin, Y.; Rigoni, E. Surrogate-assisted multicriteria optimization: Complexities, prospective solutions, and business case. *J. Multi-Criteria Decis. Anal.* **2017**, *24*, 5–24. [CrossRef]
17. Roy, P.C.; Deb, K. High Dimensional Model Representation for Solving Expensive Multi-objective Optimization Problems. In Proceedings of the 2016 IEEE Congress on Evolutionary Computation (CEC), Vancouver, BC, Canada, 24–29 July 2016.
18. Rahat, A.A.M.; Everson, R.M.; Fieldsend, J.E. Alternative Infill Strategies for Expensive Multi-objective Optimisation. In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '17), Berlin, Germany, 15–19 July 2017; ACM: New York, NY, USA, 2017; pp. 873–880. [CrossRef]
19. Gómez, R.H.; Coello, C.A.C. A Hyper-heuristic of Scalarizing Functions. In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '17), Berlin, Germany, 15–19 July 2017; ACM: New York, NY, USA, 2017; pp. 577–584. [CrossRef]
20. Deb, K.; Hussein, R.; Roy, P.C.; Toscano, G. A Taxonomy for Metamodeling Frameworks for Evolutionary Multi-Objective Optimization. *IEEE Trans. Evol. Comput.* **2018**, *23*, 104–116. [CrossRef]
21. Hussein, R.; Roy, P.C.; Deb, K. Switching between Metamodeling Frameworks for Efficient Multi-Objective Optimization. In Proceedings of the 2018 IEEE Symposium Series on Computational Intelligence (SSCI), Bangalore, India, 18–21 November 2018; pp. 1188–1195.
22. Viana, F.A.C.; Haftka, R.T.; Watson, L.T. Efficient global optimization algorithm assisted by multiple surrogate techniques. *J. Glob. Optim.* **2013**, *56*, 669–689. [CrossRef]
23. Chugh, T.; Jin, Y.; Miettinen, K.; Hakanen, J.; Sindhya, K. A Surrogate-Assisted Reference Vector Guided Evolutionary Algorithm for Computationally Expensive Many-Objective Optimization. *IEEE Trans. Evol. Comput.* **2018**, *22*, 129–142. [CrossRef]
24. Zhao, D.; Xue, D. A multi-surrogate approximation method for metamodeling. *Eng. Comput.* **2011**, *27*, 139–153. [CrossRef]
25. Bhattacharjee, K.; Singh, H.; Ray, T. Multi-Objective Optimization Using an Evolutionary Algorithm Embedded with Multiple Spatially Distributed Surrogates. *Am. Soc. Mech. Eng.* **2016**, *138*, 135–155.
26. Wang, H.; Jin, Y.; Doherty, J. Committee-Based Active Learning for Surrogate-Assisted Particle Swarm Optimization of Expensive Problems. *IEEE Trans. Cybern.* **2017**, *47*, 2664–2677. [CrossRef]
27. Gaspar-Cunha, A.; Vieira, A. A Multi-Objective Evolutionary Algorithm Using Neural Networks to Approximate Fitness Evaluations. *Int. J. Comput. Syst. Signal* **2005**, *6*, 18–36.
28. Rosales-Perez, A.; Coello, C.A.C.; Gonzalez, J.A.; Reyes-Garcia, C.A.; Escalante, H.J. A hybrid surrogate-based approach for evolutionary multi-objective optimization. In Proceedings of the IEEE Congress on Evolutionary Computation (CEC-2013), Cancun, Mexico, 20–23 June 2013; pp. 2548–2555.

29. Akhtar, T.; Shoemaker, C.A. Efficient Multi-Objective Optimization through Population-based Parallel Surrogate Search. *arXiv* **2019**, arXiv:1903.02167v1.
30. Chugh, T.; Sindhya, K.; Hakanen, J.; Miettinen, K. A survey on handling computationally expensive multiobjective optimization problems with evolutionary algorithms. *Soft Comput.* **2019**, *23*, 3137–3166. [CrossRef]
31. Isaacs, A.; Ray, T.; Smith, W. An evolutionary algorithm with spatially distributed surrogates for multiobjective optimization. In Proceedings of the 3rd Australian Conference on Progress in Artificial Life, Gold Coast, Australia, 4–6 December 2007; Springer: Berlin/Heidelberg, Germany, 2007; pp. 257–268.
32. Habib, A.; Singh, H.K.; Chugh, T.; Ray, T.; Miettinen, K. A Multiple Surrogate Assisted Decomposition-Based Evolutionary Algorithm for Expensive Multi/Many-Objective Optimization. *IEEE Trans. Evol. Comput.* **2019**, *23*, 1000–1014. [CrossRef]
33. Pan, L.; He, C.; Tian, Y.; Wang, H.; Zhang, X.; Jin, Y. A Classification Based Surrogate-Assisted Evolutionary Algorithm for Expensive Many-Objective Optimization. *IEEE Trans. Evol. Comput.* **2018**, *23*, 74–88. [CrossRef]
34. Chafekar, D.; Shi, L.; Rasheed, K.; Xuan, J. Multiobjective GA optimization using reduced models. *IEEE Trans. Syst. Man Cybern. Part C Appl. Rev.* **2005**, *35*, 261–265. [CrossRef]
35. Peitz, S.; Dellnitz, M. A Survey of Recent Trends in Multiobjective Optimal Control—Surrogate Models, Feedback Control and Objective Reduction. *Math. Comput. Appl.* **2018**, *23*, 30. [CrossRef]
36. Thoman, J.; Eichfelder, G. Trust-Region Algorithm for Heterogeneous Multiobjective Optimization. *SIAM J. Optim.* **2019**, *29*, 1017–1047. [CrossRef]
37. Banholzer, S.; Beermann, D.; Volkwein, S. POD-Based Error Control for Reduced-Order Bicriterial PDE-Constrained Optimization. *Annu. Rev. Control* **2017**, *44*, 226–237. [CrossRef]
38. Díaz-Manríquez, A.; Toscano, G.; Barron-Zambrano, J.H.; Tello-Leal, E. A Review of Surrogate Assisted Multiobjective Evolutionary Algorithms. *Comput. Intell. Neurosci.* **2016**, *2016*, 9420460. [CrossRef] [PubMed]
39. Deb, K.; Agrawal, S.; Pratap, A.; Meyarivan, T. A fast and Elitist multi-objective Genetic Algorithm: NSGA-II. *IEEE Trans. Evol. Comput.* **2002**, *6*, 182–197. [CrossRef]
40. Singh, P.; Rossi, M.; Couckuyt, I.; Deschrijver, D.; Rogier, H.; Dhaene, T. Constrained multi-objective antenna design optimization using surrogates. *Int. J. Numer. Model.* **2017**, *30*, e2248. [CrossRef]
41. Koziel, S.; Bekasiewicz, A.; Szczepanski, S. Multi-objective design optimization of antennas for reflection, size, and gain variability using Kriging surrogates and generalized domain segmentation. *Int. J. RF Microw. Comput. Eng.* **2018**, *28*, e21253. [CrossRef]
42. Beck, J.; Friedrich, D.; Brandani, S.; Fraga, E.S. Multi-objective optimisation using surrogate models for the design of VPSA systems. *Comput. Chem. Eng.* **2015**, *82*, 318–329. [CrossRef]
43. Liao, X.; Li, Q.; Yang, X.; Zhang, W.; Li, W. Multi-objective optimization for crash safety design of vehicles using stepwise regression model. *Struct. Multidiscip. Optim.* **2008**, *35*, 561–569. [CrossRef]
44. Sreekanth, J.; Datta, B. Multi-objective management of saltwater intrusion in coastal aquifers using genetic programming and modular neural network based surrogate models. *J. Hydrol.* **2010**, *393*, 245–256. [CrossRef]
45. Arias-Montaño, A.; Coello, C.A.C.; Mezura-Montes, E. Multi-objective airfoil shape optimization using a multiple-surrogate approach. In Proceedings of the 2012 IEEE Congress on Evolutionary Computation (CEC-2012), Brisbane, Australia, 10–15 June 2012; pp. 1–8.
46. D'Angelo, S.; Minisci, E.A. Multi-objective evolutionary optimization of subsonic airfoils by Kriging approximation and evolution control. In Proceedings of the IEEE Congress on Evolutionary Computation (CEC-2005), Scotland, UK, 2–5 September 2005; pp. 1262–1267.
47. Alvarado-Iniesta, A.; Cuate, O.; Schütze, O. Multi-objective and many objective design of plastic injection molding process. *Int. J. Adv. Manuf. Technol.* **2019**, *102*, 3165–3180. [CrossRef]
48. Knowles, J.; Hughes, E.J. Multiobjective Optimization on a Budget of 250 Evaluations. In *Evolutionary Multi-Criterion Optimization*; Springer: Berlin/Heidelberg, Germany, 2005; pp. 176–190.
49. Hüsken, M.; Jin, Y.; Sendhoff, B. Structure optimization of neural networks for evolutionary design optimization. *Soft Comput.* **2005**, *9*, 21–28. [CrossRef]
50. Pilát, M.; Neruda, R. Improving many-objective optimizers with aggregate meta-models. In Proceedings of the 2011 11th International Conference on Hybrid Intelligent Systems (HIS), Melacca, Malaysia, 5–8 December 2011; pp. 555–560. [CrossRef]
51. Le, M.N.; Ong, Y.S.; Jin, Y.; Sendhoff, B. A Unified Framework for Symbiosis of Evolutionary Mechanisms with Application to Water Clusters Potential Model Design. *IEEE Comput. Intell. Mag.* **2012**, *7*, 20–35. [CrossRef]
52. Li, F.; Cai, X.; Gao, L. Ensemble of surrogates assisted particle swarm optimization of medium scale expensive problems. *Appl. Soft Comput.* **2019**, *74*, 291–305. [CrossRef]
53. Jin, C.; Qin, A.K.; Tang, K. Local ensemble surrogate assisted crowding differential evolution. In Proceedings of the 2015 IEEE Congress on Evolutionary Computation (CEC-2015), Sendai, Japan, 25–28 May 2015; pp. 433–440.
54. Deb, K. *Multi-Objective Optimization Using Evolutionary Algorithms*; John Wiley & Sons: Hoboken, NJ, USA, 2001.
55. Srinivas, N.; Deb, K. Multi-Objective function optimization using non-dominated sorting genetic algorithms. *Evol. Comput. J.* **1994**, *2*, 221–248. [CrossRef]
56. Jones, D.R.; Schonlau, M.; Welch, W.J. Efficient Global Optimization of Expensive Black-Box Functions. *J. Glob. Optim.* **1998**, *13*, 455–492. [CrossRef]

57. Deb, K. An efficient constraint handling method for genetic algorithms. *Comput. Methods App. Mech. Eng.* **2000**, *186*, 311–338. [CrossRef]
58. Wierzbicki, A.P. The use of reference objectives in multiobjective optimization. In *Multiple Criteria Decision Making Theory and Application*; Springer: Berlin/Heidelberg, Germany, 1980; pp. 468–486.
59. Das, I.; Dennis, J.E. Normal-Boundary Intersection: A New Method for Generating the Pareto Surface in Nonlinear Multicriteria Optimization Problems. *SIAM J. Optim.* **1998**, *8*. [CrossRef]
60. Deb, K.; Jain, H. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints. *IEEE Trans. Evol. Comput.* **2014**, *18*, 577–601. [CrossRef]
61. Zitzler, E.; Thiele, L. Multiobjective optimization using evolutionary algorithms—A comparative case study. In Proceedings of the Conference on Parallel Problem Solving from Nature (PPSN V), Amsterdam, The Netherlands, 27–30 September 1998; pp. 292–301.
62. Coello, C.A.C.; Sierra, M.R. A study of the parallelization of a coevolutionary multi-objective evolutionary algorithm. In *MICAI 2004: Advances in Artificial Intelligence*; Springer: Berlin/Heidelberg, Germany, 2004; pp. 688–697.
63. Kendall, M.G. A new measure of rank correlation. *Biometrika* **1938**, *30*, 81–93. [CrossRef]
64. Miettinen, K. *Nonlinear Multiobjective Optimization*; Kluwer: Dordrecht, The Netherlands, 1999.
65. Roy, P.C.; Blank, J.; Hussein, R.; Deb, K. Trust-region Based Algorithms with Low-budget for Multi-objective Optimization. In Proceedings of the Genetic and Evolutionary Computation Conference Companion (GECCO '18), Kyoto, Japan, 15–19 July 2018; ACM: New York, NY, USA, 2018; pp. 195–196.
66. Alexandrov, N.M.; Dennis, J.E.; Lewis, R.M.; Torczon, V. A trust-region framework for managing the use of approximation models in optimization. *Struct. Optim.* **1998**, *15*, 16–23. [CrossRef]
67. Tian, Y.; Cheng, R.; Zhang, X.; Jin, Y. PlatEMO: A MATLAB Platform for Evolutionary Multi-Objective Optimization. *IEEE Comput. Intell. Mag.* **2017**, *12*, 73–87. [CrossRef]
68. Allmendinger, R.; Knowles, J. 'Hang on a minute': Investigations on the effects of delayed objective functions in multiobjective optimization. In *Evolutionary Multi-Criterion Optimization*; Purshouse, R.C., Fleming, P.J., Fonseca, C.M., Greco, S., Shaw, J., Eds.; Springer: Berlin/Heidelberg, Germany, 2013; pp. 6–20.
69. Blank, J.; Deb, K. *Constrained Bi-objective Surrogate-Assisted Optimization of Problems with Heterogeneous Evaluation Times: Expensive Objectives and Inexpensive Constraints*; Technical Report COIN Report 2020019; COIN Laboratory, Michigan State University: East Lansing, MI, USA, 2020.