Adaptive Switching Strategy for Metamodeling Based Multi-objective Optimization: Part II, Simultaneous and Combined Frameworks

Proteek Chandan Roy, Rayan Hussein, and Kalyanmoy Deb, *Fellow, IEEE*

COIN Report Number 2019002

***Abstract*—Most practical optimization problems are comprised of multiple conflicting objectives and constraints which involve time-consuming simulations. Construction of metamodels from a few high-fidelity solutions and then an optimization of metamodels to find infill solutions in an iterative manner stay as common metamodeling based optimization strategies. The authors have previously proposed a taxonomy of 10 different metamodeling frameworks, each of which metamodels objectives and constraints independently or in an aggregate manner. Of 10 frameworks, five proposed 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. In Part I, we have proposed an adaptive switching based metamodeling (G-ASM) method involving generative frameworks only. Motivated by the success of G-ASM on 18 different two to five-objective problems, we develop a simultaneous ASM or (S-ASM) method by switching among five simultaneous frameworks in successive epochs. Of the five frameworks, M3-2 and M4-2 frameworks are discussed for the first time here. On the same 18 problems, S-ASM performs better than the individual simultaneous frameworks alone. Then, a more efficient switching strategy (GS-ASM) involving all 10 frameworks is developed and is found to outperform both G-ASM and S-ASM. Finally, GS-ASM is compared with three other recently proposed multi-objective metamodeling methods and superior performance of GS-ASM is observed.**

***Keywords*—Surrogate model, Metamodel, Evolutionary multi-objective optimization, Kriging, Taxonomy.**

I. Introduction

PRACTICAL problems often require expensive simulation of accurate models. To get close to the optimum of these models, most multi-objective optimization algorithms need to utilize a substantially large number of solution evaluations. However, in practice, only a handful of solution evaluations are allowed due to the overall time constraint assigned to solve such problems. Researchers usually resorts to surrogate models or metamodels constructed from a few high-fidelity solution evaluations to replace the computationally expensive models to drive the optimization task. For example, Gaussian process model, Kriging, or response surface method is commonly used. The Kriging methods is specially used, since it is able to provide an approximated function and also an error estimate of the approximation [1].

Authors are with Michigan State University, East Lansing, MI 48824, USA e-mail: {kdeb,husseinr,royprote}@egr.msu.edu, see http://www.coin-laboratory.com.

In extending the metamodeling concept to multi-objective optimization problems, an obvious issue arises: multiple objective and constraint functions need to be metamodeled. Despite this challenge of multiple metamodeling efforts, a good number of studies have been made to solve computationally expensive multi-objective optimization problems using metamodeling based evolutionary algorithms [2]–[6]. In most of the studies, only unconstrained multi-objective optimization problems are addressed. Extending an unconstrained optimization algorithm to constrained optimization is not trivial [7]. The procedure employed by most of the studies are as follows. Using an initial archive of solutions obtained by the Latin-hypercube sampling, a metamodel for each function and constraint is built independently. Then, an evolutionary multi-objective optimization (EMO) algorithm is used to optimize the metamodeled objectives and constraints to find one or more infill points. Thereafter, the infill points are evaluated using high-fidelity models and saved into the archive. Next, new metamodels are built using the archive members and the procedure is repeated until all the maximum number of solution evaluations is consumed [2], [8]. We termed this basic approach as M1-2 framework in our taxonomy. It turns out that there are nine other frameworks possible in which metamodels are constructed for different combinations of objectives and constraints alone or aggregated. For example, all constraint violations can be combined to form a constraint violation function. Instead of metamodeling each constraint independently, one constraint violation function can be metamodeled. Similarly, instead of metamodeling each objective function, an aggregated objective function, such as a weighted-sum of objectives or the achievement scalarization function, or the Tchebyshev function [9], can be metamodeled. Although such an aggregation reduces the number of metamodels to be constructed for optimization, each metamodel may require a large number of high-fidelity solutions to have a small metamodeling error. In principle, each of the 10 proposed frameworks makes a balance of these two aspects by requiring $(M + J)$ metamodels to a single metamodel involving M objectives and J constraints.

It is also interesting to realize that each such framework becomes an efficient choice for certain objective and constraint landscape combinations. Since an optimization process goes through differing landscapes from start to finish, it is a natural proposition to use a switching based metamodeling strategy. In successive epochs, the most suitable framework can be identified by performing a statistical test on the current archive and adopting it for the optimization task in the next epoch. In Part I of this two-part paper, we have involved five of the 10 frameworks, which aimed at finding a single Pareto-optimal solution at a time. The generative adaptive switching based metamodeling (G-ASM) method was found to be superior to individual frameworks on 18 two to five-objective constraint and unconstrained problems.

In this paper, we extend the G-ASM concept to the remaining five frameworks which aim at finding multiple Pareto-optimal solutions simultaneously. These frameworks were termed as M1-2, M2-2, M3-2, M4-2 and M6. In M1-2, objectives and constraints are metamodeled independently and an EMO procedure is used to find multiple trade-off solutions of the metamodels. In M2-2, objectives are metamodeled independently, but constraints are combined into a single constraint violation function and metamodeled. Again, an EMO procedure is used to find multiple trade-off solutions. In M3-2, objectives are aggregated into multiple parameterized functions, which are then metamodeled separately. Constraints are metamodeled independently. A multi-modal based genetic algorithm (GA) considers multiple metamodeled functions simultaneously to find multiple trade-off solutions. M4-2 does a similar aggregation for objectives, but constraints are converted into a single constraint violation function and metamodeled. In M6, all objectives and constraints are combined in a single selection function, which is then metamodeled. Then, a niched GA is used to find multiple trade-off solutions. The resulting simultaneous ASM or S-ASM method switches among these five frameworks by means of a statistical test in successive epochs. S-ASM is then applied to the same 18 problems. Next, a more efficient GS-ASM method is developed by switching among all 10 generative and simultaneous frameworks and applied to 18 problems of this study. Finally, in order to examine the efficacy of the proposed GS-ASM method, it is compared with three recently-proposed multi-objective metamodeling methods.

In the remainder of the paper, Section II briefly describes recent related works. Section III provides a brief description of each simultaneous framework for surrogate-assisted optimization. The proposed adaptive switching (S-ASM) method is described in Section IV. Our extensive results on unconstrained and constrained test problems for each framework and S-ASM are presented in Section V. Detailed results of combined switching strategy (GS-ASM) among all ten frameworks are presented in Section VI. Finally, GS-ASM is compared with three recent algorithms in Section VII. We summarize our study of switching framework based surrogate-assisted optimization with future research directions in Section VIII.

II. Recent Metamodeling Methods for Multi-objective Optimization

The metamodeling frameworks discussed in this section attempt to solve the following multi- or many-objective optimization problem, 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 ($\mathbf{f}$):

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

In this study, it is assumed 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 $\mathbf{x}$.

A number of efficient metamodeling frameworks have been proposed recently for multi-objective optimization [10]–[14]. These frameworks use different metamodeling methods to approximate objective and constraint functions, such as radial basis functions, Kriging, Bayesian neural network, support vector regression, and others. We gave a brief description of each method in Part I; here we only mention the three recent methods that are used in this study.

Zhang et al. [8] proposed an MOEA/D-EGO algorithm which metamodeled each objective 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 solution in each optimization task. No constraint handling procedure was suggested. Thus, this method falls under our M1-2 framework.

Chugh et al. [11] 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 falls under our M1-2 framework. No constraint handling method is suggested.

Pan et al. [15] 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. This algorithm falls in M3-2 framework, more details of which are provided in this paper.

Authors proposed a taxonomy of 10 metamodeling frameworks [16] by utilizing metamodels for individual or aggregate objective and constraint functions. The 10 frameworks can be divided to two categories: generative and simultaneous frameworks. In Part I of this two-part papers, we have considered five generative frameworks, each of which aims to find a single Pareto-optimal solution in each metamodeling-cum-optimization task. A parameterized aggregation function is metamodeled and solved repeatedly with different parameter values. The taxonomy described five other frameworks, each of which aims to find multiple Pareto-optimal solutions in metamodeling-cum-optimization task. In this paper, we consider these five *simultaneous* frameworks and propose an adaptive switching strategy among them to create an efficient algorithm. In the following section, we first describe these five simultaneous frameworks and then propose the simultaneous adaptive switching based metamodeling (S-ASM) method.

III. Simultaneous Frameworks

In these frameworks, objectives, constraints, or their aggregate functions are metamodeled using a number of high-fidelity solutions from an archive, but a multi-objective or a multi-modal optimization algorithm is applied to find not one, but multiple, non-dominated infill solutions simultaneously. The infill solutions are then evaluated using high-fidelity models and included in the archive. New metamodels are the re-constructed and re-optimized to find multiple new infill solutions. This procedure is continued until the maximum allowed number of high-fidelity solutions are evaluated. In the following subsections, we describe all five simultaneous metamodeling frameworks.

A. Frameworks M1-2 and M2-2

Frameworks M1-2 and M2-2 construct an independent metamodel for each objective function. First, the original objective functions ($f_i(\mathbf{x})$ for $i = 1, \ldots, M$) are evaluated for the initial archive of N_0 solutions, created using a Latin hypercube sampling method in the entire variable search space. Then, each objective function is normalized using minimum and maximum objective values to obtain $\underline{f}_i$ for $i = 1, \ldots, M$. The normalized objective values are then used to metamodel corresponding objective functions independently to obtain $\widetilde{\underline{f}}_i(\mathbf{x})$ for $i = 1, \ldots, M$. However, the treatment of constraints is different in M1-2 and M2-2. First, each constraint function, $g_j(\mathbf{x})$ for $j = 1, \ldots, J$, is normalized using the standard method outlined in [17] to obtain the normalized constraint function $\underline{g}_j(\mathbf{x})$. In M1-2, each normalized constraint function is metamodeled separately to obtain $\widetilde{\underline{g}}_j(\mathbf{x})$ for $j = 1, \ldots, J$. Having M objective metamodels and J constraint metamodels, an evolutionary multi-objective optimization (EMO) can be used to find H (pre-specified) non-dominated solutions of the metamodeled problem. In this paper, we use NSGA-II procedure [18] for two-objective problems, and NSGA-III [19] for three or more objective problems, for this purpose. This concludes an epoch. All H infill 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.

On the other hand, in M2-2 framework, the normalized constraint functions are used to construct an aggregated constraint violation (ACV) function, as follows:

$$\mathrm{ACV}(\mathbf{x}) = \begin{cases} \sum_{j=1}^{J} \underline{g}_j(\mathbf{x}), & \text{if } \mathbf{x} \text{ is feasible,} \\ \sum_{j=1}^{J} \langle \underline{g}_j(\mathbf{x}) \rangle, & \text{otherwise.} \end{cases} \tag{2}$$

Here, the bracket operator $\langle \alpha \rangle$ is α, if $\alpha > 0$; and zero, otherwise. Thus, for all feasible archive solutions, ACV is negative and for every infeasible archive solution, ACV is positive. In M2-2, the above ACV function is metamodeled to obtain $\widetilde{\mathrm{ACV}}(\mathbf{x})$. Like in M1-2, multiple objective function metamodels ($\widetilde{\underline{f}}_i$ for $i = 1, \ldots, M$) and a single constraint violation metamodel ($\widetilde{\mathrm{ACV}}(\mathbf{x})$ are utilized during the EMO algorithm to find H non-dominated infill solutions. They are then included in the archive and the procedure is continued until $\mathrm{SE}_{\max}$ high-fidelity evaluations are completed. Thus, M1-2 constructs a total of $(M + J)$ metamodels and M2-2 constructs $(M + 1)$ metamodels in each epoch.

B. Frameworks M3-2 and M4-2

The original taxonomy study [16] suggested the possibility of two simultaneous metamodeling frameworks – M3-2 and M4-2, but this is the first time, we describe them in detail here.

In these two frameworks, M normalized objective functions ($\underline{f}_i$ for $i = 1, \ldots, M$) are combined to form a parameterized aggregate function. In this study, we construct the following achievement scalarizing function for a given $\mathbf{z}$-vector, although any other scalarizing function [9] can also be chosen:

$$\mathrm{ASF}_{34}(\mathbf{x}, \mathbf{z}) = \max_{i=1}^{M} \left(\underline{f}_i(\mathbf{x}) - z_i \right). \tag{3}$$

The reference point $\mathbf{z}$ is chosen as one of the H Das and Dennis [20] vector set ($\mathbf{Z}$). ASF values for all archive members are metamodeled to obtain H different $\widetilde{\mathrm{ASF}}_{34}(\mathbf{x}, \mathbf{z})$ functions, one for each $\mathbf{z}$-vector. All H metamodels are simultaneously used during the optimization process, as will be discussed in the next paragraph. M3-2 and M4-2 frameworks differ in the way the constraints are handled. In M3-2, each normalized constraint is metamodeled separately to obtain $\widetilde{\underline{g}}_j(\mathbf{x})$ for $j = 1, \ldots, J$, as in M1-2. However, in M4-2, the ACV function, described in Equation 2, is metamodeled to obtain $\widetilde{\mathrm{ACV}}(\mathbf{x})$. Thus, M3-2 uses $H + J$ metamodels and M4-2 uses $H + 1$ metamodels in each epoch. These are usually larger than $M + J$ and $M + 1$ metamodels needed for M1-2 and M2-2, respectively, but due to the focus of metamodels along each $\mathbf{z}$-direction in the objective space, the accuracy of the metamodels in M3-2 and M4-2 are likely to be better. Thus, M3-2 constructs a total of $(H + J)$ metamodels and M4-2 constructs $(H + 1)$ metamodels in each epoch.

We now describe the multi-modal optimization algorithm used in finding H infill points simultaneously. First, it is important to recognize that the metamodels of aggregation function of objectives and constraints are optimized, and not the original objectives and constraints. The multi-modal RGA (MM-RGA) starts with a random population of size N and uses a trust-region concept described in Part I [21]. Then, 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 solution as follows. First, a *fitness* is assigned to each population member $\mathbf{x}$ by computing $\widetilde{\mathrm{ASF}}_{34}(\mathbf{x}, \mathbf{z})$ for all H $\mathbf{z}$-vectors and then assigning the smallest value as fitness. In the binary tournament selection operation, two population members are chosen at random. The feasibility of each population member is checked based on individual constraint violations ($\widetilde{\underline{g}}_j(\mathbf{x})$ for $j = 1, \ldots, J$) for M3-2, or based on aggregate constraint violation ($\widetilde{\mathrm{ACV}}(\mathbf{x})$) for M4-2. Then, the constrained tournament operation [17] is applied to choose the winner as a parent for the subsequent recombination operation. In the tournament selection, if one member is feasible and other is not, then the former is chosen. If both are infeasible, then the one with smaller effective constraint violation, defined as follows, is chosen:

$$\mathrm{ECV}(\mathbf{x}) = \begin{cases} \sum_{j=1}^{J}\langle \underset{\sim}{\widetilde{g}}_j(\mathbf{x})\rangle, & \text{for M3-2,} \\ \langle \widetilde{\mathrm{ACV}}(\mathbf{x})\rangle, & \text{for M4-2.} \end{cases} \quad (4)$$

But, if both are feasible, then the one with smaller fitness value is chosen. Two parents are then participated in a recombination operator using the SBX operator [22] and the created child solutions are modified by using the polynomial mutation [23] operator. After N offspring population members are thus created, they are combined with the parent population P_t, and a survival selection operator is applied to pick N better solutions from $2N$ combined population members, as follows. First, the number of solutions (n_t) lying inside all the trust-regions is counted. If n_t is smaller or equal to N, all n_t members are copied to P_{t+1}, and the remaining slots of P_{t+1} are filled by choosing the nearest members to each trust-region from the rest of the combined population. On the other hand, if the n_t is larger than N, then, the best solution to each $\mathbf{z}$-vector is 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. Like in M3-1 and M4-1 frameworks described in Part I, the two trust radii are updated with the generation counter. After τ generations are completed, one best solution for each $\mathbf{z}$-vector from the final population of N members is chosen as infill solutions for inclusion in the archive. To find the best solution for a $\mathbf{z}$-vector, all N members are evaluated to compute following survival selection function:

$$\mathrm{SSF}(\mathbf{x},\mathbf{z}) = \begin{cases} \widetilde{\mathrm{ASF}}_{34}(\mathbf{x},\mathbf{z}), & \text{if } \mathbf{x} \text{ is feasible,} \\ \mathrm{ASF}_{34,\max,\mathbf{z}} + \mathrm{ECV}(\mathbf{x}), & \text{otherwise,} \end{cases} \quad (5)$$

where $\mathrm{ASF}_{34,\max,\mathbf{z}}$ is the maximum $\widetilde{\mathrm{ASF}}_{34}(\mathbf{x},\mathbf{z})$ value of all feasible members of final population of MM-RGA. Then, the solution $\mathbf{x}$ with the smallest F-value is chosen as infill point for the specific $\mathbf{z}$-vector. In the event of already picked infill solution, the second-best solution according to SSF-value is chosen and so on. Thus, at the end of the MM-RGA procedure, exactly H infill solutions are obtained. The overall frameworks for M3-2 and M4-2 are outlined in Algorithm 1.

Algorithm 1: Frameworks M3-2 and M4-2.

```
Input : Objectives: $[f_1,\ldots,f_m]^T$, Constraints: $[g_1,\ldots,g_J]^T$, $n$ (variables), Initial sample size
        $N_0$, $\mathrm{SE}_{\max}$ (maximum high-fidelity solution evaluations), MM-RGA (multi-modal
        evolutionary algorithm) with population size $N$, generations for model optimization $\tau$ and other
        parameters $\Gamma$, Reference vector set $\mathbf{Z}$, ASF (scalarization function), ACV (constraint
        violation function)
Output: Solution $\mathrm{P}_T$
1  $t \leftarrow 0$;
2  $\mathrm{P}_t, \mathrm{F}_t, \mathrm{G}_t \leftarrow \emptyset$;
3  $\mathrm{P}_{new} \leftarrow \mathrm{LHS}(\rho, n)$// Initial solutions
4  $e \leftarrow |\mathrm{P}_{new}|$;
5  while true do
       // high-fidelity evaluation of objectives
6      $\mathrm{F}_{new} = \{f_i(\mathrm{P}_{new}),\ \forall i \in \{1,\ldots,M\}\}$;
       // high-fidelity evaluation of constraints
7      $\mathrm{G}_{new} = \{g_j(\mathrm{P}_{new}),\ \forall j \in \{1,\ldots,J\}\}$;
       // merge to archive
8      $\mathrm{P}_{t+1}, \mathrm{F}_{t+1}, \mathrm{G}_{t+1} \leftarrow$
         $(\mathrm{P}_t \cup \mathrm{P}_{new}), (\mathrm{F}_t \cup \mathrm{F}_{new})$ and $(\mathrm{G}_t \cup \mathrm{G}_{new})$;
9      $e \leftarrow e + |\mathrm{P}_{new}|$// total evaluations
10     break if $e \geq \mathrm{SE}_{max}$// termination
       // construct metamodels
11     $\widetilde{\mathrm{ASF}}_{34}(\mathrm{P}_{t+1},\mathbf{z}) \leftarrow \textsc{Metamodel}(\mathrm{ASF}_{34}(\mathrm{P}_{t+1},\mathbf{z}))$
        , $\forall \mathbf{z} \in \mathbf{Z}$;
12     if M3-2 then
13         $\underset{\sim}{\widetilde{g}}_j(\mathrm{P}_{t+1}) \leftarrow \textsc{Metamodel}(\underset{\sim}{g}_j(\mathrm{P}_{t+1})),\ \forall j \in$
             $\{1,\ldots,J\}$;
14     else if M4-2 then
15         $\widetilde{\mathrm{ACV}}(\mathrm{P}_{t+1}) \leftarrow \textsc{Metamodel}(\mathrm{ACV}(\mathrm{P}_{t+1}))$;
       // Optimize model space
16     $\mathrm{P}_{new} \leftarrow$ MM-RGA$(\underset{\sim}{\widetilde{g}}_{j\in\{1,\ldots,J\}}$ or $\widetilde{\mathrm{ACV}}, N, \tau, \Gamma)$;
17     if $|P_{t+1}| + |P_{new}| > SE_{max}$ then
18         $P_{new} \leftarrow$ Randomly pick $\mathrm{SE}_{max} - |P_{t+1}|$
            solutions from $P_{new}$;
19     $t \leftarrow t + 1$;
20 return $\mathrm{P}_T \leftarrow$ filter the best solutions from $\mathrm{P}_{t+1}$
```

C. Framework M6

Framework M6 is unique and uses an intellectually challenging concept [4]. It constructs a single metamodel for each epoch by combining all M objectives and J constraints. Since multiple Pareto-optimal solutions are the target, the single metamodeled function must be multi-modal, in which each optimum corresponds to a distinct Pareto-optimal solution. The following ASF-based multi-modal function is constructed for this purpose:

$$\mathrm{ASF}_6(\mathbf{x}) = \min_{\mathbf{z}\in\mathbf{Z}} \max_{i=1}^{M} \left(\underset{\sim}{f}_i(\mathbf{x}) - z_i \right). \quad (6)$$

Then, the following selection function is constructed:

$$\mathcal{S}_6(\mathbf{x}) = \begin{cases} \mathrm{ASF}_6(\mathbf{x}), & \text{if } \mathbf{x} \text{ is feasible,} \\ \mathrm{ASF}_{6,\max} + \mathrm{CV}(\mathbf{x}), & \text{otherwise,} \end{cases} \quad (7)$$

where $\mathrm{ASF}_{6,\max}$ 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. CV($\mathbf{x}$) is same as ACV($\mathbf{x}$), except that for a feasible $\mathbf{x}$, CV is zero. Interestingly, M6 constructs a single metamodel $\widetilde{\mathcal{S}}_6(\mathbf{x})$ in each epoch. It is clear that this function is multi-modal and may have discontinuities at the constraint boundary points, thereby requiring a relatively large number of points to metamodel the function with any reasonable accuracy. Due to the complexity involved in the $\mathcal{S}_6$-function, we build a neural network model of this selection function, instead of a Kriging model used in other frameworks of this study.

We now describe the optimization procedure to solve the above selection function. Due to the multi-modal nature of $\widetilde{\mathcal{S}}_6$, we modify a standard RGA to a *niche-based* RGA (N-RGA), which is different from MM-RGA used for M3-2 and M4-2 frameworks. In MM-RGA, each population member was evaluated for all H metamodels and then associated with a specific $\mathbf{z}$-vector corresponding to the smallest $\widehat{\text{ASF}}_{34}$ value. In M6, there is only metamodeled function $\widetilde{\mathcal{S}}_6$, and there is no way to associate a population member with any $\mathbf{z}$-vector to compare it against its other niche members. But, we use a trick here to understand its niche from its variable vector. First, all archive members are clustered into H niches based on their association with its closest $\mathbf{z}$-vector (using ASF value). Then, for a new population member, first its Euclidean distance in the variable space is computed from all archive members, and then the population member is considered to be in the same niche as that of its closest archive member. For creating an offspring, two randomly chosen population members from P_t are compared based on their $\widetilde{\mathcal{S}}_6$ values and the one with the smaller value is chosen. Two such parents are then mated by the SBX operator and mutated by the polynomial mutation operator to create two child solutions. After N child solutions are created, they are combined with the parent population and a survival selection operator is applied to choose N solutions for P_{t+1}. It is similar to the survival selection operator described for M3-2 and M4-2, but instead of associating a population member based on $\mathbf{z}$-vectors, the association here is achieved using a member's niche in the variable space. After τ generations of N-RGA are performed, one *best* infill solution from each niche is picked for inclusion in the archive, as follows. First, all final N population members of N-RGA are associated with one of the H niches based on Euclidean distance in the variable space. For every niche, among all niche members, the one having the smallest $\widetilde{\mathcal{S}}_6$ value is selected. In the event of a chosen solution for a niche being already picked for a different niche earlier, the solution having the next smallest $\widetilde{\mathcal{S}}_6$ value is chosen. The procedure is continued until a distinct solution from each niche is selected.

IV. Proposed Simultaneous Adaptive Switching based Metamodeling (S-ASM) Method

Each of these five simultaneous frameworks (M1-2, M2-2, M3-2, M4-2 and M6) can be applied from the first to the last epoch of an optimization run, but as discussed in Part I, such an application may not be the most efficient approach. This is due to the fact that the intermediate points involved in an optimization process go through different complexities and a different framework of combining objectives and constraints may turn out to be more efficient for the metamodeling purpose. Moreover, in such a case, the allowable $\text{SE}_{\max}$ gets distributed among five frameworks, thereby making each framework to use a smaller number of high-fidelity evaluations. In Part I, an adaptive switching among all generative frameworks and the fundamental simultaneous framework M1-2, was able to produce better performance compared to any of the six standalone frameworks. In this study, we execute a similar adaptive switching strategy, but using all five simultaneous frameworks.

In this section, we describe the adaptive switching strategy involving a few key concepts. At the start of each epoch, new infill solutions are either available from Latin hypercube sampling or from an optimization task of metamodeled objectives and constraints, or their aggregates. After the new infill solutions are evaluated using high-fidelity models and combined with previous archive of high-fidelity solutions, the whole archive is used to evaluate and choose one of the best-performing frameworks for the next epoch. Such a task must involve a suitable performance metric and must be statistics based, which we discuss next.

We use the *selection error probability* or SEP to compare performance of different frameworks. The SEP metric was defined in Part I for generative frameworks. For completeness, we make a brief description here. At the beginning of an epoch, let us say that the archive contains N_t points, each of which is evaluated using high-fidelity models. In a 10-fold cross-validation process, 90% of N_t are used to construct metamodels based on a framework's plan, as described above. In M1-2, this means that all M objectives and all J constraints are metamodeled independently. In M6, it means constructing a single metamodel of the selection function $\widetilde{\mathcal{S}}_6$. The remaining 10% of N_t points are now ready to be evaluated using the constructed metamodel(s). Two points (p and q) from the remaining $n' = 0.1N_t$ archive points can be compared based on the metamodel(s) and superiority of one over the other can be established. Since these two points were already evaluated using high-fidelity models, their actual relationship based on the framework's plan does not cost expensive computations. If the predicted relationship (p constraint-dominates [23] q, or q constraint-dominates p, or p and q are non-constraint-dominated) matches with their actual relationship, we indicate the selection error function $E(p,q)$ to be zero, else, it is one. All pairs of (p,q) are considered one at a time and the following SEP metric is defined:

$$\text{SEP} = \frac{1}{n'} \sum_{p=1}^{n'-1} \sum_{q=p+1}^{n'} E(p,q). \tag{8}$$

A framework with a smaller SEP is considered to be better. The above process is repeated 10 times by using different blocks of 90% archive points and 10 different SEP values are obtained for each framework. Notice that these 10-fold cross-validation procedure does not require any new solution evaluations, as the whole computations are performed based on the archive points, which were already evaluated using high-fidelity computational procedures. Thereafter, the best framework ($\mathcal{M}_b$) is identified based on the smallest median SEP value of all participating frameworks. Finally, the Wilcoxon rank-sum test is performed between $\mathcal{M}_b$ and each other framework. All frameworks within a statistical insignificance (having $p > 0.05$) are identified as best-performing frameworks. Then, a random framework from these best-performing frameworks including $\mathcal{M}_b$ is chosen for the next epoch. Since each of these best-performing frameworks performs similar to $\mathcal{M}_b$ in a median sense, the choice of a random framework helps to use a diverse metamodeling landscape for the search from one epoch to another, thereby prohibiting the overall method to not get stuck with similar search behaviors. A pseudo-code of the proposed S-ASM method is provided in Algorithm 2.

Algorithm 2: Adaptive Swithing Framework

```
Input  : Objectives: $[f_1, \dots, f_m]^T$, Constraints:
         $[g_1, \dots, g_J]^T$, $n$ (variables), Initial sample size
         $N_0$, SE$_{max}$ (maximum high-fidelity solution
         evaluations), Swithing frameworks $\mathcal{M}_i$ for
         $i \in \{1 \dots, S\}$ where $S$ is the number of
         frameworks, parameters and functions of each
         framework $\Gamma_i$ for $i \in \{1 \dots, s\}$, Number of
         solutions per epoch $u$, Number of partitions for
         cross-validation $K$
Output: P$_T$
$t \leftarrow 0$;
P$_t$, F$_t$, G$_t \leftarrow \emptyset$;
P$_{new} \leftarrow$ LHS$(\rho, n)$// Initial solutions
$e \leftarrow |$P$_{new}|$;
while True do
    // high-fidelity evaluation of objectives
    $F_{new} = \{f_i(\text{P}_{new}),\ \forall i \in \{1, \dots, M\}\}$;
    // high-fidelity evaluation of constraints
    $G_{new} = \{g_j(\text{P}_{new}),\ \forall j \in \{1, \dots, J\}\}$;
    // merge to archive
    P$_{t+1}$, F$_{t+1}$, G$_{t+1} \leftarrow$
      (P$_t \cup$ P$_{new}$), (F$_t \cup$ F$_{new}$), (G$_t \cup$ G$_{new}$);
    $e \leftarrow e + |$P$_{new}|$// total evaluations
    break if $e \geq$ SE$_{max}$// termination
    Calculate $\{$ASF(.), 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 $P_{t+1}$, $\forall k \in \{1, \dots, K\}$;
    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 $Q^k_{t+1}$;
            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 SE$_{max} - |P_{t+1}|$
          solutions from $P_{new}$;
    $t \leftarrow t + 1$;
    // end of epoch
return P$_T \leftarrow$ filter best solutions from P$_{t+1}$
```

A summary of metamodeled functions and the optimization algorithms used to optimize metamodeled functions for all 10 frameworks is provided in Table I. M3-1 and M3-2 require to construct the maximum number of metamodels among all the frameworks and M6 requires the least. All generative frameworks require H independent applications of a single-objective optimization algorithm (RGA) and all simultaneous frameworks employ an EMO or an multi-modal RGA once in every epoch.

TABLE I: 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	$(\underline{f}_1, \dots, \underline{f}_M)$ $(\underline{g}_1, \dots, \underline{g}_J)$	$M + J$	RGA	H
M1-2	Same as above	$M + J$	NSGA-II	1
M2-1	$(\underline{f}_1, \dots, \underline{f}_M)$ & ACV	$M + 1$	RGA	H
M2-2	Same as above	$M + 1$	NSGA-II	1
M3-1	ASF$_{34}$ & $(\underline{g}_1, \dots, \underline{g}_J)$	$H + J$	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

V. Results And Discussion of S-ASM Method

First, we present the results of the S-ASM method on 18 different test and engineering problems used in Part I. The problems include two to five-objective, constrained and unconstrained problems. S-ASM results are then compared with five individual frameworks. Thereafter, all 10 frameworks (generative and simultaneous) are included as options for switching, thereby making the overall GS-ASM method more robust. The performances of three switching methods (G-ASM, S-ASM, and GS-ASM) are compared. Finally, the three switching methods are compared with three recently suggested multi-objective metamodeling methods: MOEA/D-EGO [8], K-RVEA [11] and CSEA [15].

A. Parameter Settings

For two-objective problems, we use NSGA-II [18] for M1-2 and M2-2 frameworks. For problems with higher number of objectives, we use NSGA-III [19] procedure. For M3-2 and M4-2, we use the MM-RGA method described in Section III. For M6, we apply niched RGA (N-RGA) method described in Section III-C. A population size (N) is 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 II. Other parameter settings are as follows: Number of generations $\tau = 300$, crossover probability $p_c = 0.95$, mutation probability $p_m = 1/n$, distribution index for SBX operator is $\eta_c = 20$, and distribution index for polynomial mutation operator $\eta_m = 20$. The number of reference points, SE$_{\text{max}}$, resulting epochs for each problem are presented in Table II.

B. Two-objective Unconstrained Problems

First, we apply our proposed methodologies to two-objective unconstrained problems: ZDT1, ZDT2, ZDT3, ZDT4 and

TABLE II: Parameter values for 18 problems.

Problem	n	M	J	N_0	$\text{SE}_{\max}$	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

TABLE III: IGD values obtained from simultaneous frameworks and S-ASM for 18 test problems are presented. Best performing framework is marked in bold and other statistically similar frameworks are marked in bold-italics with their p-values in the second row.

Problem	M1-2	M2-2	M3-2	M4-2	M6	S-ASM
ZDT1	***0.0055***	-	***0.0054***	-	0.0134	**0.0051**
	p=0.5011	-	p=0.6992	-	p=8.1e-5	-
ZDT2	**0.0006**	-	0.0091	-	0.7237	0.0009
	-	-	p=8.1e-5	-	p=8.1e-5	p= 0.0025
ZDT3	**0.0021**	-	0.1905	-	0.0832	***0.0044***
	-	-	p=8.1e-5	-	p=8.1e-5	p=0.475
ZDT4	5.4345	-	**0.4345**	-	6.1551	***0.5179***
	p=0.0002	-	-	-	p=0.0002	p= 0.5545
ZDT6	0.4836	-	0.4716	-	**0.2133**	***0.4082***
	p=8.1e-5	-	p=8.1e-5	-	-	p= 0.0569
OSY	***0.1881***	22.9999	4.7767	18.3376	57.1587	**0.1750**
	p=0.7427	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	-
TNK	**0.0008**	0.0285	0.0112	0.0374	0.0399	***0.0009***
	-	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=0.4307
SRN	***1.0093***	**0.9261**	1.5136	1.4887	2.4171	***1.1589***
	p= 0.3933	-	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=0.0762
BNH	***0.0463***	**0.0446**	0.3287	0.3660	0.7130	***0.0452***
	p=0.9476	-	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=0.8438
WB	***0.2316***	0.8475	1.3825	3.4077	1.4111	**0.1744**
	p=0.2372	p= 0.0039	p=0.0012	p=8.1e-5	p=8.1e-5	-
DTLZ2	**0.0334**	-	0.0504	-	0.0774	***0.0334***
	-	-	p=8.1e-5	-	p=8.1e-5	p=0.7928
C2DT-LZ2	***0.0336***	-	**0.0319**	-	0.0441	***0.0327***
	p= 0.3932	-	-	-	p=8.1e-5	p=0.772
CAR	0.5012	**0.2982**	0.4049	0.4425	0.5357	***0.3893***
	p=8.1e-5	-	p=0.0003	p=8.1e-5	p=8.1e-5	p=0.7928
DTLZ5	**0.0095**	-	0.0154	-	0.0542	***0.0109***
	-	-	p=8.1e-5	-	p=8.1e-5	p=0.5114
DTLZ4	***0.0906***	-	0.1257	-	***0.0873***	**0.0801**
	p=0.1860	-	p= 0.0151	-	p=0.2122	-
DTLZ7	***0.0766***	-	1.2630	-	0.8299	**0.0721**
	p=0.3579	-	p=8.1e-5	-	p=8.1e-5	-
DTLZ2-5	**0.0398**	-	0.1443	-	0.1103	***0.0536***
	-	-	p=8.1e-5	-	p=8.1e-5	p=0.6250
C2DT-LZ2-5	**0.0368**	-	0.1170	-	0.2084	***0.0488***
	-	-	p=8.1e-5	-	p=8.1e-5	p=0.7110

ZDT6. Table III presents the median IGD values of 11 runs for each framework. In the absence of any constraint or having a single constraint, M1-2 and M2-2 are identical frameworks; so are M3-2 and M4-2. Thus, we keep a blank for M2-2 and M4-2 for unconstrained and single-constraint problems. But for S-ASM, we consider every framework to be different, although in principle, the pairs will have a similar performance with the SEP metric. It is clear from the table that S-ASM performs better or equivalent to four out of five ZDT problems, whereas M1-2 performs best in two problems. While M3-2 performs well in ZDT1 and ZDT4, M6 performs the best in ZDT6 problem.

The epoch-wise proportion of usage of each framework over 11 runs is shown in Figure 1 for all five ZDT problems. Except in ZDT6, M1-2 (and M2-2) turns to be a dominating framework on other four ZDT problems. In ZDT6, M3-2 (and M3-4) shows its dominance. In ZDT4, M1-2 and M3-2 methods are found to be switching between them early on, but settles with M1-2 at the latter part of the optimization runs. Interestingly, in ZDT1, a switch form M1-2 to M6 at the end is observed. The switching of frameworks for the median performing run for ZDT1, ZDT4 and ZDT6 is shown in Figure 2. It is interesting to observe that in this particular run, M6 is found to consistently perform the best from third epoch on ZDT1. In ZDT4, S-ASM alternates between four frameworks to produce better result than M1-2 alone. The dominance of different frameworks from epoch to epoch is clear from these plots. More plots are provided in the supplementary document.

C. Two-objective Constrained Problems

Next, we apply five frameworks and S-ASM to two-objective constrained problems: BNH, SRN, TNK, OSY [23] and the welded beam problem (WB). S-ASM performs the best on OSY and WB, followed by M1-2. Other individual frameworks do not perform at all on these two problems. In all problems, the performance of S-ASM and M1-2 is better or comparable to other individual frameworks. Figure 3 shows the epoch-wise proportion of usage of different frameworks in 11 runs. The plots for OSY, TNK, BNH, and SRN make it clear that the S-ASM almost always chooses M1-2 as the best-performing framework, as supported by IGD values in Table III.

D. Three and More Objective Constrained and Unconstrained Problems

Next, we apply all six methods 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 [19]). Table III shows that while M2-2 works the best on CAR, M3-2 on C2-DTLZ2, and M6 on DTLZ4, the performance of S-ASM is the best with statistical significance, followed by M1-2.

On two five-objective unconstrained DTLZ2 and constrained C2-DTLZ2 problems, M1-2 performs the best, followed by S-ASM method with statistically insignificant difference in performance between the two.

Table V calculates the rank of each of the six methods in solving 18 problems. The table shows that S-ASM performs

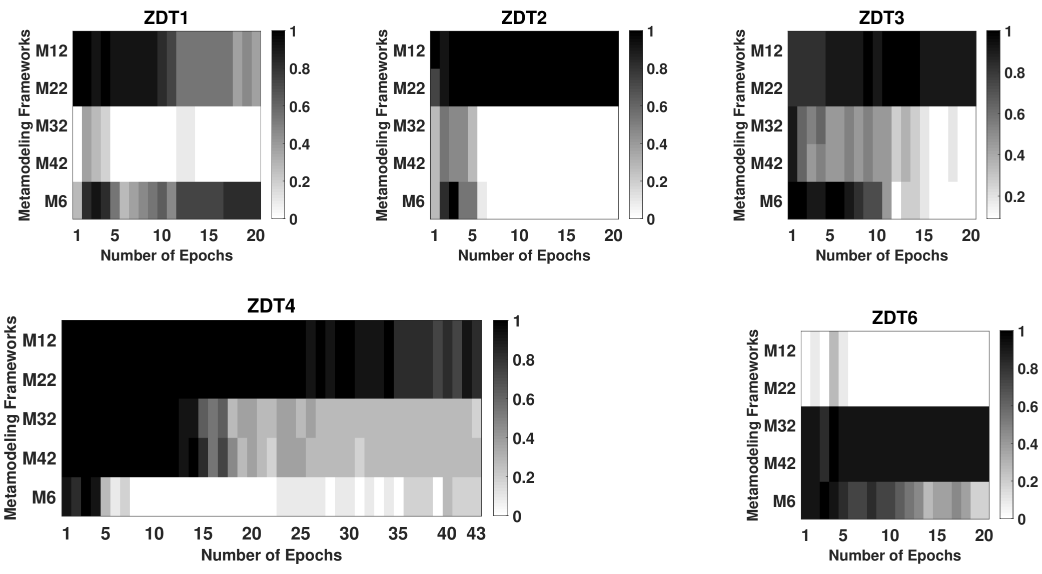

Fig. 1: Epoch-wise proportion of usage of five frameworks in 11 runs for ZDT problems.

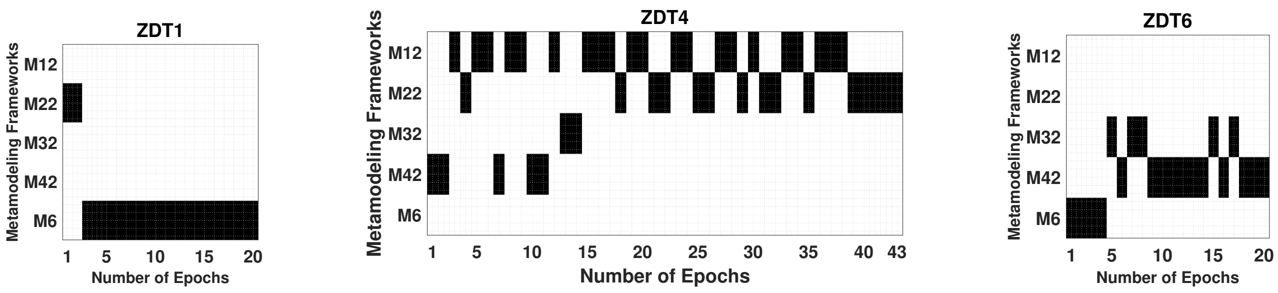

Fig. 2: Switching among frameworks for the median IGD run of S-ASM for ZDT1, ZDT4 and ZDT6.

the best overall, followed by M1-2 and then M2-2, indicating that metamodeling of objectives independently is a better approach for simultaneous frameworks. M6, with a single metamodel, performs the worst.

VI. Combined Generative and Simultaneous Adaptive Switching based Metamodeling (GS-ASM) Method

In Part I, we have observed G-ASM's superior performance compared to five individual generative frameworks. In the previous section, we have observed an overall superior performance of S-ASM on all 18 problems compared to five simultaneous frameworks. Thus, it is better to use the proposed adaptive switching strategy than individual frameworks, but in both studies, the framework M1-2 came close to the switching methods. Next, we would like to make a more extensive study in which our adaptive switching method can choose any one of 10 frameworks. Since at the beginning of each epoch, H new infill solutions are created, either one at a time by the generative frameworks or simultaneously by the approaches of this paper, a transition from a generative framework to a simultaneous framework is possible. We call this the generative-simultaneous or GS-ASM method.

Table V presents the IGD value of all 10 frameworks on 18 all problems. On all ZDT problems, GS-ASM is always one of the top performing methods, except in CAR. Table VI shows the average rank of 10 frameworks and GS-ASM. When each of the 10 frameworks and GS-ASM are ranked according to Wilxozon rank-sum test, the GS-ASM performs the best overall, followed by M1-2 and then M2-2.

Non-dominated solutions of the final archive of G-ASM, S-ASM and GS-ASM (marked as SW1, SW2 and SW3, respectively) are shown for ZDT problems in Figure 4. For ZDT1 and ZDT2 problems, all three switching methods are able to find solutions on the Pareto-optimal front in only 500 SEs. For ZDT3, ZDT4 and ZDT6, GS-ASM performs better overall than other two swithcing methods, as expected. Figure 5 plots the epoch-wise proportion of usage of different

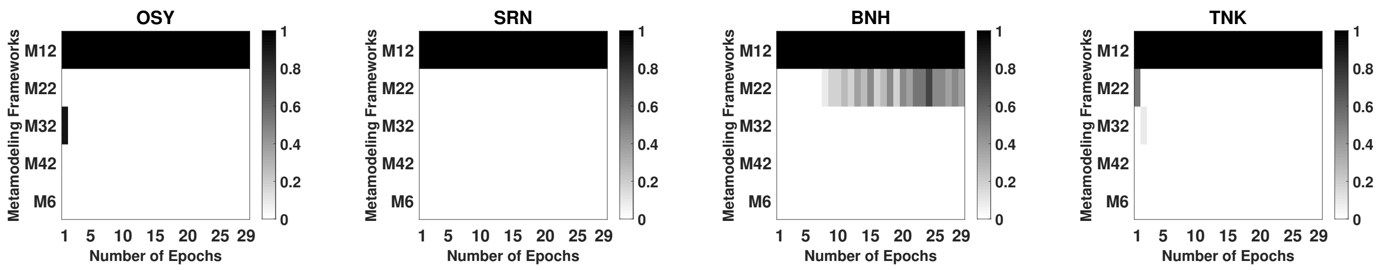

Fig. 3: Epoch-wise proportion of usage of simultaneous frameworks in 11 runs of S-ASM for two-objectives constrained problems.

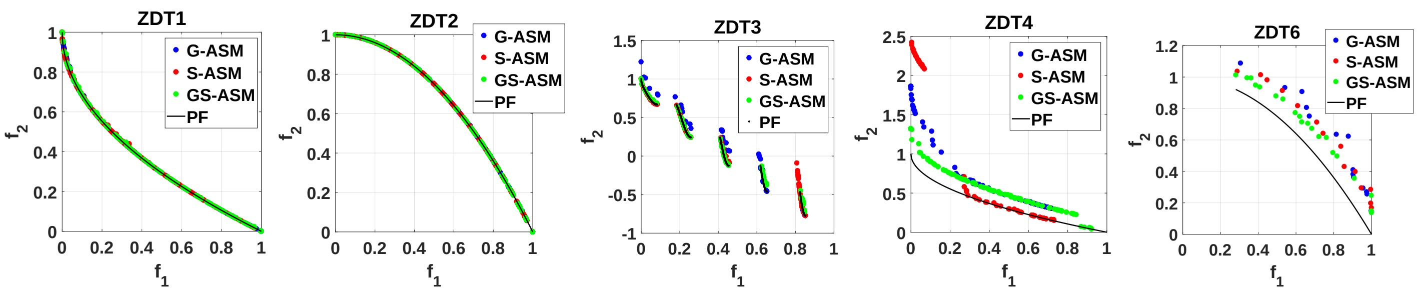

Fig. 4: Obtained non-dominated solutions of the final archive for the median G-ASM, S-ASM, and GS-ASM runs for ZDT problems.

TABLE IV: Rank of five simultaneous frameworks and S-ASM for 18 problems.

Problem	M1-2	M2-2	M3-2	M4-2	M6	S-ASM
ZDT1	1	1	1	1	6	1
ZDT2	1	1	4	4	6	3
ZDT3	1	1	5	5	3	1
ZDT4	4	4	1	1	6	1
ZDT6	3	3	5	5	1	1
OSY	1	5	3	4	6	1
TNK	1	4	3	5	6	1
SRN	1	1	5	4	6	1
BNH	1	1	4	5	6	1
WB	1	3	4	6	5	1
DTLZ2	1	1	4	4	6	1
C2DTLZ2	1	1	1	1	6	1
CAR	5	1	3	4	6	1
DTLZ5	1	1	4	4	6	1
DTLZ4	1	1	5	5	1	1
DTLZ7	1	1	5	5	4	1
DTLZ2-5	1	1	5	5	4	1
C2DTLZ2-5	1	1	4	4	6	1
Average	1.50	1.78	3.67	4.00	5.05	**1.11**

frameworks in 11 runs for three ZDT problems. Switching among different frameworks is clear from these plots, More such analysis can be found in the supplementary document on other problems.

Non-dominated solutions for the two-objective constrained problems are shown in Figure 6. With only 800 or 1,000 SEs, remarkable performance of all three switching methods is demonstrated here.

The epoch-wise proportion of usage of 10 frameworks in 11 runs are shown in Figure 7 for three and five-objective problems. It can be clearly seen that M3-1 to M6 frameworks are not chosen by the GS-ASM method on most of these problems. 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.

VII. Comparative Studies

Next, we examine the performance of three adaptive switching metamodeling strategies by comparing them with a few recent algorithms, namely, MOEA/D-EGO [8], K-RVEA [11], and CSEA [15]. Algorithms are implemented in PlatEMO [24]. Since these three competing algorithms can only be applied to unconstrained problems, only ZDT and DTLZ problems are considered here. Identical parameters settings as those used with ASM methods are used for the three competing algorithms. Table VII presents the mean IGD value of each algorithm. The Wilcoxon rank-sum test results are also shown. It is clearly evident that GS-ASM method outperforms three competing methods, of which K-RVEA performs well only on two of the nine problems.

Finally, we compare the performance of all three switching methods on all 18 problems in Table VIII. Rank of a method is shown in brackets. It is clear that GS-ASM performs better

TABLE V: IGD values obtained from all the individual frameworks and proposed combined switching algorithm for test problems are presented. Best performing framework and other statistically similar frameworks are marked in bold with their p-values in the second row.

Problem	M1-1	M2-1	M1-2	M2-2	M3-1	M4-1	M3-2	M4-2	M5	M6	GS-ASM
ZDT1	**0.00090**	-	***0.00555***	-	***0.00447***	-	***0.00537***	-	-	0.01337	***0.00130***
	-	-	p= 0.4701	-	p= 0.4702	-	p=0.7928	-	-	p=8.1e-5	p=0.091
ZDT2	***0.00065***	-	***0.00062***	-	0.00568	-	0.00910	-	-	0.72366	**0.00055**
	p=0.2372	-	p=0.2372	-	p=8.1e-5	-	p=8.1e-5	-	-	p=8.1e-5	-
ZDT3	0.06778	-	**0.00212**	-	0.17123	-	0.19050	-	-	0.08315	***0.00391***
	p=8.1e-5	-	-	-	p=8.1e-5	-	p=8.1e-5	-	-	p=8.1e-5	p=0.369
ZDT4	**0.28900**	-	5.43450	-	***0.29300***	-	0.43450	-	-	6.15510	***0.39992***
	-	-	p=8.1e-5	-	p=0.4307	-	p=0.0126	-	-	p=8.1e-5	p=0.1310
ZDT6	***0.37058***	-	0.48360	-	***0.24192***	-	0.47159	-	-	**0.21327**	***0.24440***
	p=0.2934	-	p=8.1e-5	-	p=0.8438	-	p=0.0013	-	-	-	p= 0.3933
OSY	***0.15323***	24.57940	0.18806	22.99990	6.26550	18.49200	4.77670	18.33760	45.18110	57.15870	**0.12110**
	p= 0.2301	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	-
TNK	0.01726	0.04383	***0.00082***	0.02849	0.01180	0.03332	0.01121	0.03743	0.03077	0.03990	**0.00080**
	p=8.1e-5	p=8.1e-5	p=0.318	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	-
SRN	**0.13191**	4.17160	1.00930	0.92614	1.06120	1.20480	1.51360	1.48870	1.28450	2.41710	***0.13406***
	-	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=0.1891
BNH	0.69434	0.74425	***0.04630***	***0.04457***	0.23728	0.23923	0.32874	0.36600	0.23699	0.71300	**0.04176**
	p=8.1e-5	p=8.1e-5	p=0.5114	p=0.5994	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	-
WB	***0.13794***	0.55529	0.23159	0.84746	***0.16909***	0.88586	1.39250	3.40770	0.96166	1.41110	**0.08960**
	p=0.2933	p=8.1e-5	p=0.0126	p=8.1e-5	p=0.1007	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	-
DTLZ2	0.07870	-	**0.03340**	-	0.05377	-	0.05040	-	-	0.07736	***0.03701***
	p=8.1e-5	-	-	-	p=8.1e-5	-	p=8.1e-5	-	-	p=8.1e-5	p=0.562
C2DTLZ2	0.05130	-	***0.03355***	-	0.03493	-	***0.03190***	-	0.12403	0.04410	**0.03062**
	p=8.1e-5	-	p= 0.115	-	p=0.008	-	p=0.148	-	p=8.1e-5	p=8.1e-5	-
CAR	0.43510	0.43145	0.50119	**0.29817**	0.39809	0.42223	0.40494	0.44251	0.50061	0.55569	0.40110
	p=8.1e-5	p=8.1e-5	p=8.1e-5	-	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5
DTLZ5	0.01960	-	**0.00948**	-	0.01352	-	0.01537	-	-	0.05421	***0.01252***
	p=8.1e-5	-	-	-	p=8.1e-5	-	p=8.1e-5	-	-	p=8.1e-5	p=0.0605
DTLZ4	**0.05840**	-	***0.09024***	-	0.20668	-	0.12570	-	-	***0.08731***	***0.07934***
	-	-	p=0.1203	-	p=8.1e-5	-	p=8.1e-5	-	-	p=0.3933	p=0.425
DTLZ7	0.89316	-	***0.07664***	-	0.87172	-	1.26300	-	-	0.82989	**0.06529**
	p=8.1e-5	-	p=0.2122	-	p=8.1e-5	-	p=8.1e-5	-	-	p=8.1e-5	-
DTLZ2-5	0.21450	-	**0.03981**	-	0.14401	-	0.14403	-	-	0.11028	***0.04918***
	p=8.1e-5	-	-	-	p=8.1e-5	-	p=8.1e-5	-	-	p=8.1e-5	p=0.595
C2DTLZ2-5	0.17341	-	***0.03676***	-	0.15388	-	0.11669	-	0.29291	0.20842	**0.03441**
	p=8.1e-5	-	p=0.8541	-	p=8.1e-5	-	p=8.1e-5	-	p=8.1e-5	p=8.1e-5	-

TABLE VI: Average rank of 10 frameworks and GS-ASM on 18 problems based in Wilcoxon rank-sum test.

M1-1	M2-1	M1-2	M2-2	M3-1	M4-1
4.94	6.55	2.88	3.00	4.33	5.27
M3-2	M4-2	M5	M6	GS-ASM	
6.11	6.88	6.16	8.44	**1.11**	

or equivalent on all 18 problems. GS-ASM performs the best on 12 problems, while S-ASM and G-ASM performs the best on three problems each.

VIII. Conclusions

In this paper, we have developed a simultaneous adaptive switching based metamodeling (S-ASM) method for solving multi-objective optimization problems. The S-ASM method switches to an appropriate simultaneous frameworks which create a pre-specified number of infill points simultaneously by using either an evolutionary multi-objective algorithm or by using a multi-modal or a niche-based real-parameter genetic algorithms. Kriging modeling method has been used in all cases. S-ASM method has been found to perform better compared to five simultaneous frameworks alone on 18, two to five-objective, constrained and unconstrained test and engineering problems.

A comparison between generative ASM, simultaneous ASM and a combination (GS-ASM) involving 10 different frameworks has revealed that GS-ASM performs the best overall. Finally, GS-ASM has been compared against three recently proposed surrogate-assisted, unconstrained, multi-objective optimization methods. On eight or nine problems, GS-ASM has been found to perform better than the three existing methods.

This and Part I papers have amply demonstrated the superiority of adaptive switching based metamodeling methods for multi-objective optimization problems. The studies can be extended to include different metamodeling methods at each epoch, such as RBF or other response surface methods. More initial samples can be allocated to complex metamodeling functions, such as for frameworks M5 and M6, so that more accurate metamodels can be obtained.

References

[1] D. R. Jones, “A taxonomy of global optimization methods based on response surfaces,” *Journal of global optimization*, vol. 21, no. 4, 2001.

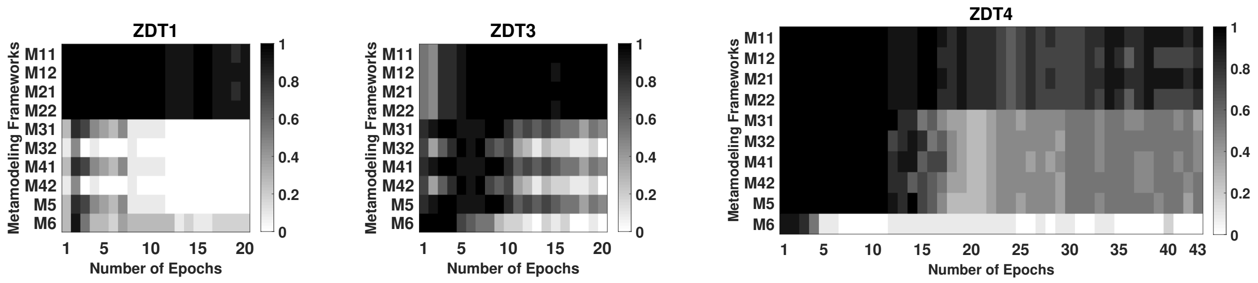

Fig. 5: Epoch-wise proportion of usage of 10 frameworks in 11 runs of GS-ASM for ZDT problems.

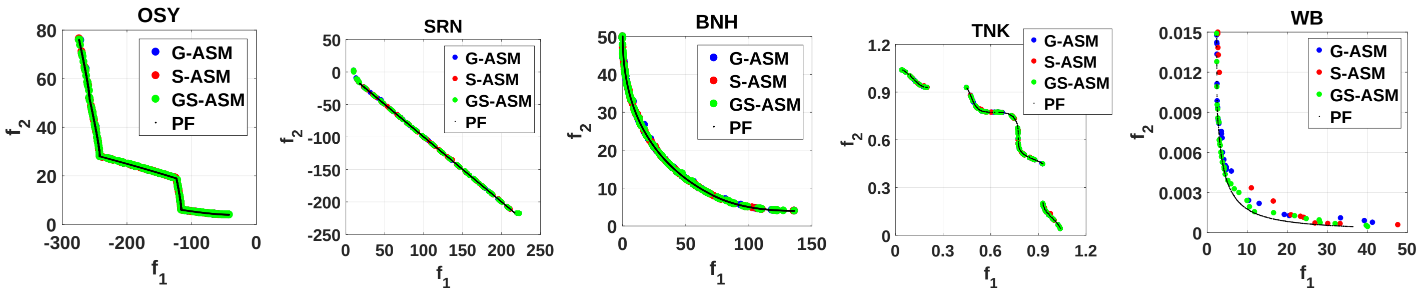

Fig. 6: Obtained non-dominated solutions of the final archive for the median G-ASM, S-ASM, and GS-ASM runs for two-objective constrained problems.

TABLE VII: Median IGD on unconstrained problems using GS-ASM and MOEA/D-EGO, K-RVEA, and CSEA algorithms.

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

[2] R. Hussein and K. Deb, "A generative kriging surrogate model for constrained and unconstrained multi-objective optimization," in *GECCO*, ACM Press, 2016.

[3] K. Deb, R. Hussein, P. Roy, and G. Toscano, "Classifying metamodeling methods for evolutionary multi-objective optimization: First results," in *Evolutionary Multi-Criterion Optimization EMO*, Springer, 2017.

[4] P. Roy, R. Hussein, and K. Deb, "Metamodeling for multimodal selection functions in evolutionary multi-objective optimization," in *GECCO*, ACM Press, 2017.

[5] K. S. Bhattacharjee, H. K. Singh, and T. Ray, "Multi-objective optimization with multiple spatially distributed surrogates," *Journal of Mechanical Design*, vol. 138, no. 9, 2016.

[6] K. S. Bhattacharjee, H. K. Singh, T. Ray, and J. Branke, "Multiple surrogate assisted multiobjective optimization using improved preselection," in *IEEE CEC*, 2016.

[7] R. H. Byrd, J. Nocedal, and R. A. Waltz, "Knitro: An integrated package for nonlinear optimization," in *Large-Scale Nonlinear Optimization*, Springer US, 2006.

[8] Q. Zhang, W. Liu, E. Tsang, and B. Virginas, "Expensive Multiobjective Optimization by MOEA/D With Gaussian Process Model," *IEEE Transactions on Evolutionary Computation*, vol. 14, june 2010.

[9] K. Miettinen, *Nonlinear Multiobjective Optimization.* Kluwer, 1999.

[10] F. A. C. Viana, R. T. Haftka, and L. T. Watson, "Efficient global optimization algorithm assisted by multiple surrogate techniques," *Journal of Global Optimization*, 2013.

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

[12] D. Zhao and D. Xue, "A multi-surrogate approximation method for metamodeling," *Engineering with Computers*, 2011.

[13] K. Bhattacharjee, H. Singh, and T. Ray, "Multi-objective optimization using an evolutionary algorithm embedded with multiple spatially distributed surrogates," *The American Society of Mechanical Engineers*, vol. 138, no. 9, pp. 135–155, 2016.

[14] H. Wang, Y. Jin, and J. Doherty, "Committee-based active learning for

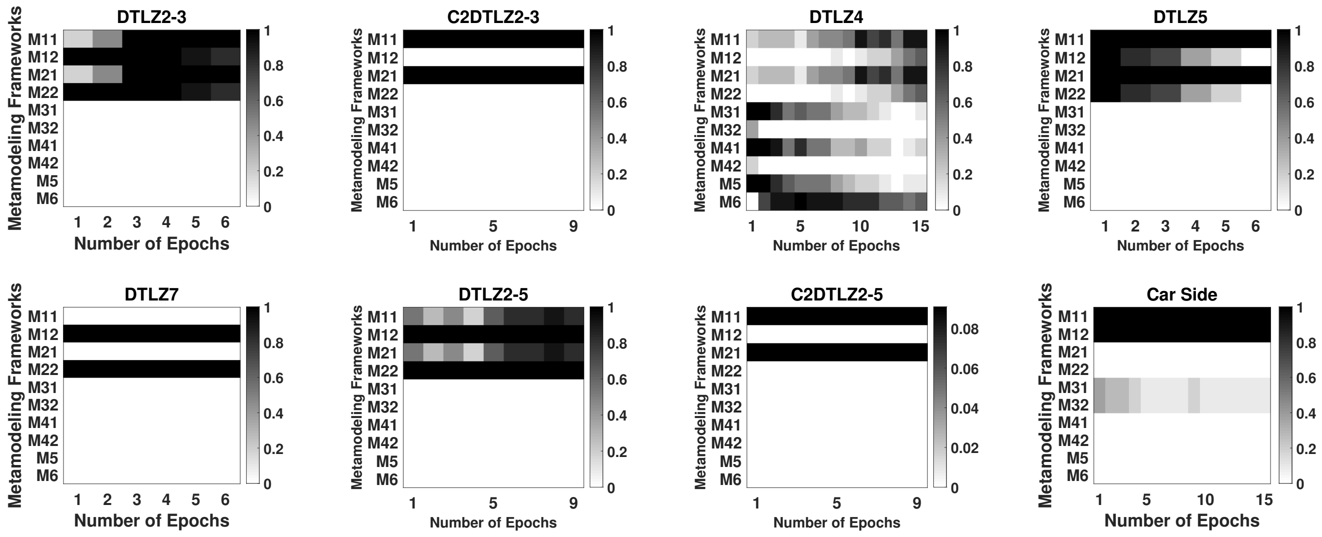

Fig. 7: Epoch-wise proportion of usage of 10 frameworks in 11 runs of GS-ASM for three and five-objective problems.

surrogate-assisted particle swarm optimization of expensive problems," *IEEE Transactions on Cybernetics*, vol. 47, pp. 2664–2677, Sept 2017.

[15] L. Pan, C. He, Y. Tian, H. Wang, X. Zhang, and Y. Jin, "A classification based surrogate-assisted evolutionary algorithm for expensive many-objective optimization," *IEEE Transactions on Evolutionary Computation*, pp. 1–1, 2018.

[16] K. Deb, R. Hussein, P. C. Roy, and G. Toscano, "A taxonomy for metamodeling frameworks for evolutionary multi-objective optimization," *IEEE Transactions on Evolutionary Computation*, 2018.

[17] K. Deb, "An efficient constraint handling method for genetic algorithms," *Computer Methods in App. Mechanics and Engg.*, 2000.

[18] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA–II," *IEEE Transactions on Evolutionary Computation*, pp. 182–197, April 2002.

[19] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints," *IEEE Transactions on Evolutionary Computation*, vol. 18, pp. 577–601, 2014.

[20] S. Das, A. Abraham, and A. Konar, "Automatic Clustering Using an Improved Differential Evolution Algorithm," *IEEE Transactions on Sys. Man Cyber. Part A*, vol. 38, pp. 218–237, Jan. 2008.

[21] P. C. Roy, J. Blank, R. Hussein, and K. Deb, "Trust-region based algorithms with low-budget for multi-objective optimization," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, GECCO '18, pp. 195–196, ACM, 2018.

[22] K. Deb and R. Agrawal, "Simulated Binary Crossover for Continuous Search Space," tech. rep., Departement of Mechanical Enginering, Indian Institute of Technology, Kanpur, India, 1994.

[23] K. Deb, *Multi-objective optimization using evolutionary algorithms*. John Wiley & Sons, 2001.

[24] Y. Tian, R. Cheng, X. Zhang, and Y. Jin, "Platemo: A MATLAB platform for evolutionary multi-objective optimization," *IEEE Computational Intelligence Magazine*, vol. 12, no. 4, pp. 73–87, 2017.

TABLE VIII: Median IGD for generative, simultaneous, and combined strategies are tabulated for all 18 problems.

Problem	G-ASM	S-ASM	GS-ASM
ZDT1	**0.00099** (1)	***0.00505*** (1)	***0.00130*** (1)
	-	p=0.9476	p=0.091
ZDT2	***0.00057*** (1)	0.00090 (3)	**0.00055** (1)
	p=0.7372	p= 0.0025	-
ZDT3	0.00456 (3)	***0.00441*** (1)	**0.00391** (1)
	p=0.092	p=0.134	-
ZDT4	***0.45803*** (1)	***0.51786*** (1)	**0.39992** (1)
	p=0.2122	p=0.9476	-
ZDT6	**0.19038** (1)	***0.40817*** (1)	***0.24440*** (1)
	-	p= 0.0569	p= 0.0612
OSY	***0.16376*** (1)	***0.17496*** (1)	**0.12110** (1)
	p= 0.4307	p= 0.317	-
TNK	0.00105 (3)	***0.00089*** (1)	**0.00080** (1)
	p=0.00008	p=0.4307	-
SRN	***0.13434*** (1)	1.15890 (3)	**0.13406** (1)
	p=0.8955	p=8.1e-5	-
BNH	0.07765 (3)	***0.04516*** (1)	**0.04176** (1)
	p=0.00008	p=0.5114	-
WB	***0.12819*** (1)	0.17440 (3)	**0.08960** (1)
	p=0.1891	p= 0.0488	-
DTLZ2	***0.03921*** (1)	**0.03344** (1)	***0.03701*** (1)
	p=0.114	-	p=0.157
C2DTLZ2	***0.03401*** (1)	***0.03269*** (1)	**0.03062** (1)
	p=0.095	p=0.122	-
CAR	***0.39060*** (1)	**0.38930** (1)	***0.40110*** (1)
	p=0.753	-	p=0.561
DTLZ5	***0.01123*** (1)	**0.01085** (1)	***0.01252*** (1)
	p=0.455	-	p=0.167
DTLZ4	***0.08012*** (1)	***0.08005*** (1)	**0.07934** (1)
	p=0.1150	p=0.1674	-
DTLZ7	***0.07261*** (1)	***0.07211*** (1)	**0.06529** (1)
	p=0.3841	p=0.4315	-
DTLZ2-5	**0.04643** (1)	***0.05361*** (1)	***0.04918*** (1)
	-	p=0.1924	p=0.7903
C2DTLZ2-5	***0.04891*** (1)	***0.04880*** (1)	**0.03441** (1)
	p= 0.0878	p=0.0712	-
Average	1.33	1.44	**1.00**