Adaptive Switching Strategy for Metamodeling Based Multi-objective Optimization: Part I, Generative Frameworks

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

COIN Report Number 2019001

***Abstract*—Evaluating computationally expensive objective and constraint functions is one of the main challenges faced when solving real-world optimization problems. For handling such problems, it is common to use a metamodeling approach. Metamodels for objectives and constraints are initially formed using a few high-fidelity solution evaluations. Then, the metamodels are optimized to find a set of in-fill solutions. New metamodels are then formed by including high-fidelity evaluations of in-fill solutions to the initial set. This procedure is continued in a progressive manner until a predetermined budget of solutions are evaluated. A recent study has provided a taxonomy of 10 different frameworks for forming metamodels of objective and constraint combinations. In this paper, we propose a novel adaptive method for switching among five different generative metamodeling frameworks and one simultaneous framework in multiple epochs. Statistical tests on multi-objective convergence and diversity-preservation metrics are made at the start of each epoch to determine one of the six frameworks which is most suitable at that instant. In the second part of this extensive research, we perform a similar study using the remaining five simultaneous frameworks, followed up by another study including all 10 frameworks. Results of this study clearly show the efficacy and efficiency of the proposed adaptive switching approach compared to past framework-wise studies and to three recently-proposed other metamodeling algorithms on challenging multi-objective optimization problems using a limited budget of high-fidelity evaluations.**

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

I. Introduction

THE use of metamodels (or surrogate models) to approximate the functional form of exact or simulated models of objective and constraint functions by using a few high-fidelity solution evaluations is a common approach [1]–[3]. Among various methods, the Kriging method is one of the widely used metamodels, which can provide an estimated function value and also simultaneously provide an error estimate of the approximation [4]–[6]. The developments in optimization methods have recently led to an increasing interest in metamodeling efforts.

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

Most metamodeling efforts in evolutionary multi-objective optimization (EMO), so far, seem to have taken a straightforward extension of single-objective metamodeling approaches [7]–[9]. Also these metamodeling efforts did not consider constraints in enough details. On some occasions, bounded variable problems are considered. But, this defeats the whole purpose of using a metamodeling approach in the first place. Metamodeling methods are needed mainly to solve practical problems, which are often computationally expensive [10], [11]. Constraints are also inevitable in practical problems. Thus, addressing metamodeling methods to solve only unconstrained or bounded-variable problems does not solve the issue completely. In most existing constraint-based metamodeling approaches, every objective and constraint function is metamodeled independently [12], [13]. Thereafter, a standard evolutionary multi-objective optimization (EMO) framework is applied to the metamodels, instead of the original objective and constraint functions, to find a non-dominated front. In some studies, this sequence of metamodeling-EMO combination is repeated multiple times so that a refinement of the metamodels can occur with iterations (we call here epochs to distinguish them from iterations used in an optimization process) [5], [14]–[17]. However, the above framework of metamodeling every objective and constraint function independently is one of at least 10 different frameworks, recently proposed by the authors [18], [19]. For example, in another framework, all constraints can be combined into a single constraint violation function, formulated by adding violations of all constraints in a normalized manner [20]. Thus, instead of metamodeling each constraint separately, the effort can be reduced substantially by metamodeling a single combined constraint violation function. In another framework, instead of metamodeling every objective function separately, a combined scalarized objective function (such as, an achievement scalarization function or a Tchebyshev function [21]–[23]) can be metamodeled. The taxonomy study by the authors [18] proposed 10 different combinations of metamodeling objectives and constraints and evaluated each of the frameworks by applying them on alone from start to finish of a simulation on a number of two, three and five-objective optimization problems. The taxonomy includes one method that requires as many as $(M + J)$ metamodels (where M and J are the number of objectives and constraints, respectively) to another extreme method that requires only one metamodel approximating a combination of all objectives and constraints as a *selection* function used in EMO algorithms.

While the metamodeling effort can be reduced by modeling a combined objective or a combined constraint violation function, the flip side is that each metamodel of the combined functions is likely to be more complex having discontinuous, non-differentiable, and multi-modal landscapes, despite their individual functional forms being simple. Thus, comparatively a large number of data points may have to be used to obtain an adequate accuracy of a combined metamodeled function. However, we argue here that the effectiveness of these different framework possibilities in an EMO algorithm largely depends on the complexity of the respective landscapes in which the current state of points lie. Thus, instead of one framework (say, the one in which every objective and constraint function is metamoded independently) demonstrating the best performance from the start to the end of a metamodeling-based optimization run, a switching among different metemodeling frameworks may produce a better performance. Since, it is not known which framework will be best when, an adaptive strategy is needed for the algorithm to determine this aspect every time a switching is desired. In this paper, we propose an adaptive switching strategy after a every few new in-fill solutions are created (ending in an epoch) to determine the best framework for the next epoch. Statistical tests are conducted with existing high-fidelity solutions using a new but more appropriate performance metric for optimization studies to choose the next best metamodeling framework. To take into account the uncertainty affecting the accuracy and the efficiency of any approximation mathematical model [24], we also implement a trust-region concept for increasing the success of newly created in-fill points.

In the remainder of the paper, Section II briefly introduces different metamodeling frameworks used in this study. This section also provides a brief description a few recent multi-objective metamodeling studies, which we have used to compare our switching strategy with. Section IV provides a description of the proposed adaptive switching strategy in detail. Section IV-C describes the trust-region concept used in our optimization approaches. Thereafter, extensive results on unconstrained and constrained test problems are presented in Section V. Finally, important conclusions of this extensive study and plausible future extensions are discussed in Section VI.

II. Frameworks for Metamodeling in Multi-objective Optimization

The metamodeling frameworks discussed in this section attempts to solve the following multi- or many-objective optimization problem, involving n real-valued variables ($\mathbf{x}$), J inequality constraints ($\mathbf{g}$) (equality constraints are assumed to be converted to two inequality constraints), and M objective functions ($\mathbf{f}$):

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

In this study, it is assumed that all objective and constraint functions are computationally expensive to compute and that they must be computed independent to each other for every new solution $\mathbf{x}$.

We provide a brief description of our recently proposed taxonomy of various metamodeling approaches for multi- and many-objective optimization algorithms [18]. The taxonomy finds 10 different broad frameworks based on the cardinality of metamodels for objectives and constraints, as illustrated in Figure 1. In this paper, we focus on five frameworks (M1-1, M2-1, M3-1, M4-1 and M5) which use a generative method of finding a single Pareto-optimal solution at a time [21], after the objectives and constraints are metamodeled. We call these frameworks as *generative* frameworks. We also include a popular simultaneous framework M2-1 in our study here.

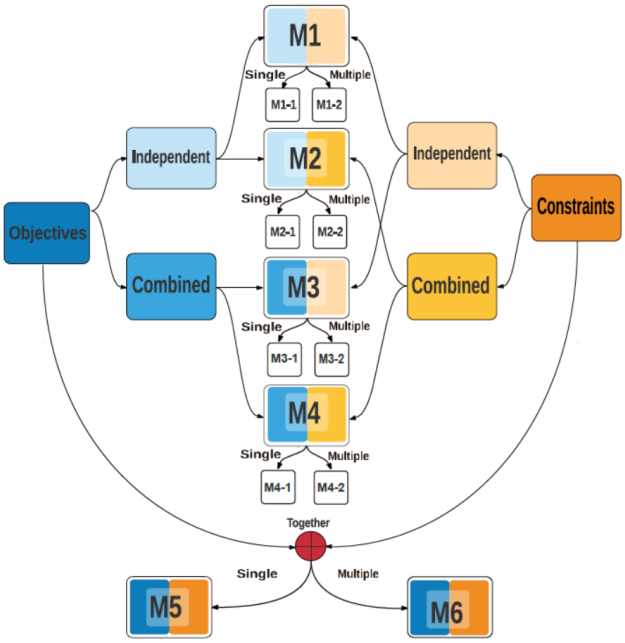

Fig. 1: The proposed taxonomy of ten different metamodeling approaches for multiple and many-objective optimization. (Taken from [18])

A. M1-1 and M2-1 Frameworks

The original taxonomy for metamodeling based multi-objective optimization [18] did not include a detailed description of M1-1 and M2-1 frameworks, which we provide here.

The metamodeling algorithm for M1-1 and M2-1 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. 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 high-fidelity evaluation of initial archive members, so that the minimum and maximum values of $\underline{f}_i(\mathbf{x})$ evaluations is zero and one, respectively. Then, metamodels are constructed for each of the M normalized objective functions independently: $(\tilde{\underline{f}}_1(\mathbf{x}), \ldots, \tilde{\underline{f}}_M(\mathbf{x})), \forall i \in \{1, 2, \ldots, M\}$ using a chosen metamodeling method. For all implementations here,

we use the Kriging metamodeling method [25]. While the objective metamodeling approach is identical to both M1-1 and M2-1, the metamodeling of constraint functions is different.

For M1-1, 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 [26], and then metamodeled separately to obtain an approximate function ($\tilde{\underline{g}}_j(\mathbf{x})$) using the same metamodeling method (Kriging method is adopted here) used for metamodeling objective functions. For M2-1, a single aggregated constraint violation function (ACV($\mathbf{x}$)) is first constructed using the normalized constraint functions, as follows:

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

where the bracket operator $\langle \alpha \rangle$ is α if $\alpha > 0$ and zero, otherwise. In M2-1, the constraint violation function is then metamodeled to obtain $\widetilde{\mathrm{ACV}}(\mathbf{x})$. Thus, it is clear that for high-fidelity solutions, ACV($\mathbf{x}$) takes a negative value for feasible solutions and a positive value for an infeasible solution.

After all $(M + J)$ or $(M + 1)$ metamodels are constructed for M1-1 or M2-1, respectively, the metamodeled normalized objectives are combined into a single aggregated function and optimized with metamodeled constraints to find a single infill point using a single-objective evolutionary optimization algorithm (real-coded genetic algorithm (RGA) [27]). In τ generations of RGA, the following achievement scalarization function ($\mathrm{ASF}_{12}(\mathbf{x}, \mathbf{z})$) [28] is optimized for every $\mathbf{z}$ vector:

$$\mathrm{ASF}_{12}(\mathbf{x}, \mathbf{z}) = \max_{j=1}^{M} \left(\tilde{\underline{f}}_j(\mathbf{x}) - z_j \right), \tag{3}$$

where the vector $\mathbf{z}$ is one of the Das and Dennis's [29] approach on the unit simplex on the M-dimensional hyperspace. Thus, for H different $\mathbf{z}$ vectors, H different ASF_{12} functions are formed and optimized one after the other. The RGA procedure is modified using a trust-region concept, which we describe in Section IV-C. The best solution for each subproblem constitutes one infill point and is sent for a high-fidelity evaluation. The solution is then included in the archive of high-fidelity solutions to obtain $\mathcal{A}_1$. After all H solutions are included in the archive, one epoch of the M1-1 or M2-1 framework optimization problem is completed. In the next epoch, all high-fidelity solutions are used to normalize the objective functions and constraints, and the above process is repeated to obtain $\mathcal{A}_2$. The process is continued until all pre-specified maximum solution evaluations ($\mathrm{SE}_{\max}$) is completed. A basic structure of methodologies M1-1 and M2-1 are outlined in Algorithm 1. Thus, in M1-1, a total of $(M+J)$ metamodels are constructed and H RGA optimization procedures are run to create H infill solutions in each epoch. The only difference in M2-1 is that a total of $(M + 1)$ metamodels are constructed in each epoch.

Algorithm 1: Metamodeling Frameworks M1-1 and M2-1.

Input : Objectives: $(f_1, \ldots, f_M)^T$, normalized constraints: $(\underline{g}_1, \ldots, \underline{g}_J)^T$, n (variables), N_0 (initial sample size), $SE_{\max}$ (total high-fidelity evaluations), RGA (real-parameter genetic algorithm), Γ (parameters of RGA), R (reference direction set), α (fraction of samples used for each reference direction), ASF (scalarization function), ACV (constrained violation function)

Output: P_T

```
P ← LHS(N_0, n) // Latin hypercube sampling
F ← N(f_m(P)), ∀m ∈ {1,...,M} // high-fidelity evaluations (Objectives) and normalize
G ← g_j(P), ∀j ∈ {1,...,J} // high-fidelity evaluations (constraints)
eval ← N_0 // number of function evaluations
while eval < SE_max do
    for r ∈ R do
        P_r ← Choose nearest α|P| solutions from r
        F̃ ← METAMODEL(f_m(P_r)), ∀m ∈ {1,...,M}
        if M1-1 then
            G̃ ← METAMODEL(g_j(P_r), ∀j ∈ {1,...,J})
        else if M2-1 then
            G̃ ← METAMODEL(ACV(G(P_r)))
        x_r ← RGA(ASF_12(F̂, r), Ĝ, Γ) // returns the best found solution along r
        F_{x_r} ← f_m(x_r), ∀m ∈ {1,...,M}// Evaluate objectives of x_r
        G_{x_r} ← g_j(x_r), ∀j ∈ {1,...,J} // Evaluate constraints of x_r
        P ← P ∪ {x_r}
        F ← N(F ∪ F_{x_r})
        G ← G ∪ G_{x_r}
        eval ← eval + 1
        if eval ≥ SE_max then
            Break out of all loops
return P_T ← Non-dominated solutions of P
```

B. M3-1 and M4-1 Frameworks

In these two methods, the normalized objective functions, $\underline{f}_i(\mathbf{x})$ for $i = 1, \ldots, M$, obtained using current high-fidelity solutions, are aggregated using the following ASF_{34} transformation:

$$\mathrm{ASF}_{34}(\mathbf{x}, \mathbf{z}) = \max_{j=1}^{M} \left(\underline{f}_j(\mathbf{x}) - z_j \right), \tag{4}$$

where $\mathbf{z}$ is defined as before. The $\mathrm{ASF}_{34}(\mathbf{x}, \mathbf{z})$ for each of a total H predefined $\mathbf{z}$-vectors is now metamodeled using a metamodeling method (Kriging used here).

In M3-1, each normalized constraint function is metamodeled separately as in M1-1. In M4-1, the constraint violation function, as described in Equation 2 and in M2-1, is constructed and metamodeled.

The trust-region based RGA is used to solve each opti-

mization subproblem. In each epoch of M3-1 framework, the metamodeled function $\widetilde{\text{ASF}}_{34}(\mathbf{x}, \mathbf{z})$ function, along with J metamodeled constraint functions, $\tilde{\underline{g}}_j(\mathbf{x})$, for $j = 1, \ldots, J$, is optimized using a RGA procedure H times, each with a predefined $\mathbf{z}$-vector. The RGA procedure uses its own constraint handling procedure to find a single near-optimal solution for each H metamodeled subproblem, thereby finding a total of H infill solutions for the next epoch. M3-1 uses a total of $H + J$ metamodels in each epoch. The only difference in M4-1 is that the metamodeled ACV function (Equation 2) is supplied to the RGA procedure, instead of J individual metamodeled constraint functions. Thus, in M4-1, a total of $H + 1$ metamodels are constructed in each epoch. Epochs are continued until total prescribed SEs are completed.

C. M5 Framework

Metamodeling frameworks M1-1 to M4-1 are straightforward extensions of single-objective metamodeling methods used in the context of evolutionary algorithms. Framework M5 proposes a direct metamodeling approach which not only reduces the cardinality of distinct metamodeling efforts, it is also algorithm specific. A metamodel of the outcome of an algorithm's selection operation is directly constructed here. The focus of M5 is to use a generative multi-objective optimization approach in which a single Pareto-optimal solution is found at a time by using a combined *selection* function involving all objective and constraint functions together, as used in a specific generative EMO algorithm. For example, in a generative version of an EMO algorithm, two solutions A and B in a population can be compared by using a constrained ASF function for a specific $\mathbf{z}$-vector:

$$\mathcal{S}_5(\mathbf{x}, \mathbf{z}) = \begin{cases} \text{ASF}_{34}(\mathbf{x}, \mathbf{z}), & \text{if } \mathbf{x} \text{ is feasible,} \\ \text{ASF}_{\max} + \langle \text{ACV}(\mathbf{x}) \rangle, & \text{otherwise.} \end{cases} \tag{5}$$

Here, the parameter $\text{ASF}_{\max}$ is the worst ASF_{34} function value of all feasible solutions of the high-fidelity solutions in the archive. Here, the ASF values are obtained using the normalized objective functions ($\underline{f}_i(\mathbf{x})$, $i = 1, \ldots, M$) as in Equation 4 and the constraint violation function $\text{ACV}(\mathbf{x})$ is obtained using the normalized constraint functions, as in Equation 2. The selection function $\mathcal{S}_5(\mathbf{x}, \mathbf{z})$ is then metamodeled using a metamodeling method (Kriging is used here) for each $\mathbf{z}$-vector to obtain $\widetilde{\mathcal{S}}_5(\mathbf{x}, \mathbf{z})$ function.

Thus, in each epoch of M5, a total of H metamodels are constructed, one for each $\mathbf{z}$-vector. The trust-region based RGA procedure is then used to optimize the unconstrained $\widetilde{\mathcal{S}}_5(\mathbf{x}, \mathbf{z})$ function to find H infill points one at a time for each $\mathbf{z}$-vector, for the next epoch. The procedure is terminated after $\text{SE}_{\max}$ are completed. Clearly, other scalarization approaches, such as weighted-sum function or epsilon-constraint function or a generic Tchebyshev function [21] can also be used to construct the selection function, instead of the ASF formulation.

It is somewhat clear that while the number of metamodels required for the construction process reduce from M1-1 ($M + J$) to M5 (H), but the respective function(s) to be metamodeled get more complex, thereby ideally requiring more high-fidelity points to obtain an accurate approximation. In this study, we use the same number of points for metamodeling functions for each framework, to not introduce further algorithmic parameters, but this remains as an interesting future work.

D. Inclusion of M1-2 Framework

Frameworks M1-1, M2-1, M3-1, M4-1 and M5 all use a generative approach in which H different $\mathbf{z}$-parameterized subproblems are metamodeled and solved independently in each epoch to find a set of H non-dominated points. However, as observed in our literature survey, the framework M1-2, in which a simultaneous multi-objective algorithm (such as, an EMO procedure including NSGA-II [30]) is used to find H non-dominated infill solutions in each generation (epoch, in our terminology) is employed to find H infill points by simultaneous optimization of M metamodeled normalized objective $\tilde{\underline{f}}_i(\mathbf{x})$, $i = 1, \ldots, M$ with J metamodeled normalized constraint $\tilde{\underline{g}}_j(\mathbf{x})$, $j = 1, \ldots, J$ functions. This approach constructs $(M + J)$ metamodels in each epoch.

III. Recent Multi-objective Surrogate Studies

In our previous studies [18], [19], we performed an extensive literature survey on studies involving metamodeling assisted EMO approaches. Since then, a few more interesting and relevant studies have come to our attention. Here we provide a brief review of them.

Zhao et al. [31] 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.

Zhang et al. [5] 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. [6] proposed a surrogate-assisted adaptive reference vectors guided evolutionary algorithm (K-RVEA) for multi-objective unconstrained optimization problems. Each objective function is metamodeled independently using a Gaussian process approach (Kriging). According to our proposed taxonomy [18], both procedures fall under the M1-2 framework.

Datta et al. [13] proposed a surrogate-assisted evolution strategy for constrained multi-objective optimization (SMES) using Radial Basis Function (RBF) to metamodel each objective and constraint function. Due to this treatment, this

method falls under our M1-2 framework. Bhattacharjee et al. [32] used multiple spatially distributed surrogates, in which each surrogate is modeled using multiple methods, such as Radial Basis Function (RBF), Kriging, and Response Surface Methodology (RSM), in different regions of the search space. Both algorithms are build on NSGA-II platform [30]. Since one surrogate model in both the above procedures is constructed for each objective and constraint function separately, they fall under our M1-2 framework.

Rahat et al. [33] proposed an approach which can either construct a scalarized objective from individual metamodels of objective functions or metamodel a scalarized objective function. They provided an user choice for this option. These methods can be classified as our M1-1 and M3-1 frameworks, respectively. Three different scalarizing functions – hypervolume improvement, dominance ranking, and minimum signed distance – are used in their study. Kriging was used as the metamodeling method, but no constraint handling strategy was proposed.

Pan et al. [34] 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 will be provided in Part-II of this study.

Nghia Le et al. [35] proposed multiple co-objective evolutionary optimization by cross-surrogate assisted memetic augmentation algorithm (CSAMA). In this algorithm, standard quadratic regression or polynomial regression as a surrogate model is employed for construction of each objective function separately. For selecting best solutions, they used NSGA-II algorithm as a global search, Thereafter, the Tchebycheff scalarization function is applied as a local search on aggregated objective function. The CSAMA algorithm is implemented for unconstrained optimization problems only. This algorithm falls under M1-2 (for the global search) and M1-1 (for the local search) frameworks according to our taxonomy.

The above descriptions indicate how our proposed taxonomy [18] includes most of the possible and existing multi-objective metamodeling approaches. We choose three of the above metamodeling-based EMO approaches – MOEA/D-EGO, K-RVEA, and CSEA – as a basis of comparison of our proposed adaptive switching approach.

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

Each metaomodeling framework in our proposed taxonomy requires to build metamodels for different individual or aggregated objective and constraint functions. 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 fair well 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 multi-objective optimization problems in our previous studies [18], [19] 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 (or six frameworks of this study) to solve each problem independently using $\mathrm{SE}_{\max}$ high-fidelity evaluations, and then determine the specific framework which performs the best using an EMO metric, such as hypervolume or IGD. This will be computationally expensive, requiring 10 (or six) times more than the prescribed $\mathrm{SE}_{\max}$. If each framework is allocated only 1/10 (or 1/6) of $\mathrm{SE}_{\max}$, they may be insufficient to find comparatively good solutions. A better approach would be use an adaptive switching strategy, in which the most suitable framework is chosen at every step of the optimization process. In this section, we propose one such adaptive switching strategy.

We call a 'step' during the optimization process for assessing different metamodeling frameworks to choose the best-performing framework as an *epoch*. In each epoch, exactly H new infill solutions are created, 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 LHS solutions, and in each subsequent epoch, H new infill members are added to the archive. At the start of t-th epoch, each of the six participating frameworks (M1-1, M1-2, M2-1, M3-1, M4-1 and M5) are used to construct its respective metamodels using all N_t archive members. Then, a 10-fold cross-validation method (described in Section IV-B) is used with a suitable performance metric (described in Section IV-A) to determine the next suitable framework for the next epoch. A flowchart of the proposed adaptive switching based metamodeling strategy is illustrated in Figure 2. In the following subsection, we describe a new performance metric used for choosing the selected framework.

A. Performance Metric for Framework Selection

We propose a *selection error probability* (SEP) metric which is appropriate for an optimization task. 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 3, 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, but when points x_1 and x_3 are compared, the prediction is wrong. Out of three pairwise comparisons, two predictions are correct and one is wrong, thereby making a selection error probability of 1/3 in this case. We argue that in an optimization procedure, it is the SEP which provides a better selection error than the actual function values. Unfortunately, many optimization methods

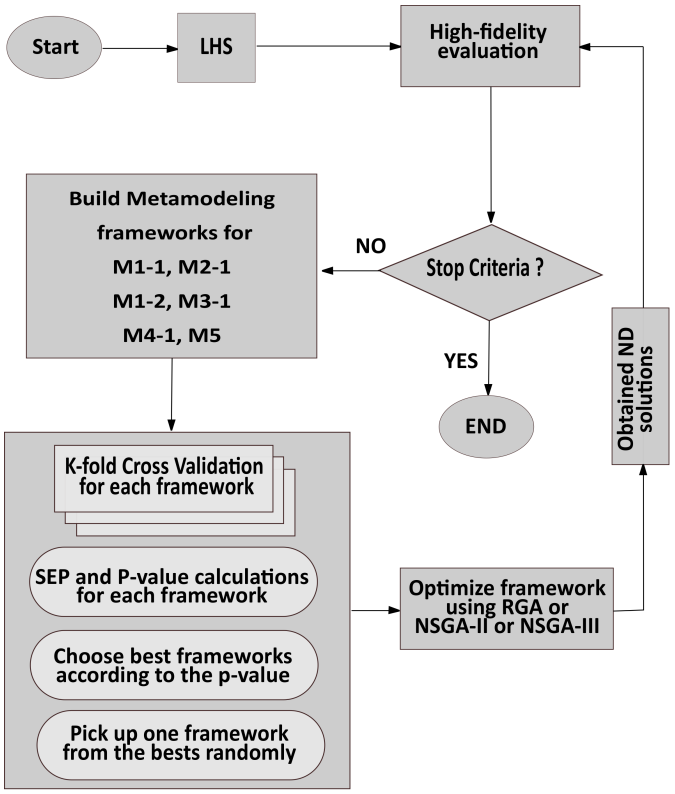

Fig. 2: Flowchart of the proposed G-ASM method.

borrow the mean squared error (MSE) metric, commonly-used in regression and metamodeling studies, but such a metric may not be the most appropriate metric for metamodeling-based approaches in optimization.

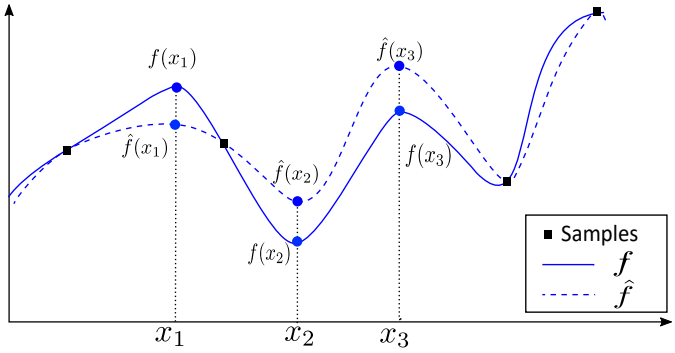

Fig. 3: Selection Error Probability (SEP) concept is illustrated.

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), 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:

$$\text{SEP} = \frac{1}{N}\sum_{p=1}^{n-1}\sum_{q=p+1}^{n} E(p,q). \tag{6}$$

The definition of a 'winner' can be easily extended to multi-objective and constrained multi-objective optimization by considering the domination [21] and constraint-domination [27] status of two points p and q.

B. Identifying a Suitable Framework for Next Epoch

A framework with a smaller SEP metric value over a 10-fold cross-validation is considered better. We describe the framework selection procedure in the following. At the end of each epoch, the new infill points are evaluated using high-fidelity evaluations and added to the archive. Then, 90% of archive points are chosen at random and respective metamodels are constructed according to the framework's plan described above. Then, the constructed metamodels are used to compare every pair (p and q) of the remaining 10% of archive points. Constrained domination checks are performed to establish the superiority of one point over the other and compared with that on the original problem. Then, the selection error function ($E(p,q)$) and finally the SEP are computed. The above process is repeated 10 times by using different blocks of 90% 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 median SEP value of 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. Then, a random framework from the statistically insignificant frameworks including $\mathcal{M}_b$ is selected for the next epoch. Since each of these frameworks performs similar to $\mathcal{M}_b$ in a median sense, the choice of a random framework helps to use a diverse metamodeling landscapes for the search from one epoch to another, thereby prohibiting the overall method to not get stuck in similar search behaviors.

C. Trust-Region Based Real-Coded Genetic Algorithms

Whenever metamodels are used instead of the original functions during an optimization task, an uncertainty of a solution's true function value always exists. Predictions of solutions close to high-fidelity solutions are more accurate than predictions far from them. Therefore, we use a trust-region method [36] in which predictions are restricted within a radius R_{trust} from each high-fidelity solution in the variable space. If R_{trust} is infinite, any solution can be accepted as an in-fill solution and the overall process will be unreliable. Contrarily, if R_{trust} is chosen to be tiny, metamodel may be well correlated with the exact model, but the search power will be lost. Thus, a suitable value of R_{trust} is desired. For multi-objective optimization, the goal is to find multiple diverse solutions. Thus, we introduce

another radius R_{prox} which defines the minimum distance any new solution should have from an existing high-fidelity archive member. Combining the above concepts, we enforce 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, we also reduce the two radii after every metamodeling task 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 below.

The RGA procedure is modified as follows. For creating an offspring solution, no trust-region is used and a standard binary constrained tournament selection is applied on two competing population members using the metamodeled objectives, constraints, or selection function to determine the better solution. After the offspring population of size N_{RGA} is created, it is combined with the parent population P_t of size N_{RGA}, and then better half is chosen for the next generation as parent population P_{t+1}. This is where the trust region concept is used. First, we count the number of solutions in the combined population inside the trust regions. If the number is smaller than or equal to N_{RGA}, 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_{RGA}, the same binary constrained tournament selection method is applied to pick N_{RGA} solutions and copied in P_{t+1}.

A summary of metamodeled functions and the optimization algorithms used to optimize metamodeled functions for all five generative frameworks and M1-2 is provided in Table I. It is clear that M3-1 needs to construct the largest number of metamodels and M2-1 the least.

TABLE I: Summary of metamodeled functions and optimization algorithms needed in each epoch for six frameworks.

Framework	Metamodeling functions	#Metamodels	Optimization method	#Opt. runs
M1-1	$(\underline{f}_1, \ldots, \underline{f}_M)$ $(\underline{g}_1, \ldots, \underline{g}_J)$	$M + J$	RGA	H
M1-2	Same as above	$M + J$	NSGA-II	1
M2-1	$(\underline{f}_1, \ldots, \underline{f}_M)$ & ACV	$M + 1$	RGA	H
M3-1	ASF_{34} & $(\underline{g}_1, \ldots, \underline{g}_J)$	$H + J$	RGA	H
M4-1	ASF_{34} & ACV	$H + 1$	RGA	H
M5	$\mathcal{S}_5$	H	RGA	H

V. Results

The RGA uses the simulated binary crossover (SBX), and polynomial mutation, with parameters as follows: Number of generations = 100, crossover probability = 0.95, mutation probability = $1/n$, distribution index for SBX operator = 1, and distribution index for polynomial mutation operator = 10. The NSGA-II procedure used in M1-2 is also applied with the same parameter values as above. For one run of RGA or NSGA-II, we use a maximum of $\tau = 50$ generations. We perform 11 runs of all metamodeling frameworks on all test problems. For unconstrained problems (ZDT) with no constraints and three-objective problem C2DTLZ2 having a single constraint, two frameworks M1-1 and M2-1 are identical. In such a case, we only show the results for M1-1. The same situation occurs for M3-1, M4-1, and M5 for unconstrained problems. For We only show the results of M3-1 for such cases. For C2DTLZ2 problems, M5 is different from M3-1 and M4-1. It is worth mentioning here, we do not make any effort to choose optimal parameter setting for each algorithm, rather identical parameters are used to provide a representative performance of each methodology. In Table II, we summarize the other parameter setting for all test problems used in this study.

TABLE II: Parameter settings used in the study.

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

A. Two-Objective Unconstrained Problems

First, we apply our proposed methodologies to two-objective unconstrained problems ZDT1, ZDT2, ZDT3, ZDT4 and ZDT6 with only $\text{SE}_{\max}$=500 or 1,000 high-fidelity evaluations. IGD values are presented in Table III. It is clear that except in ZDT4, in other four problems, the generative adaptive switching based metamodeling (G-ASM) method performs either equivalent or better than individual metamodeling frameworks. In ZDT2 and ZDT6, G-ASM performs the best in terms of median performance. Interestingly, M1-1 as a single metamodeling framework performs well in three of the five ZDT problems, but G-ASM performs well in four of the five problems. M1-2 and M3-1 performs well in two of the five problems. Non-dominated solutions for the median run of the G-ASM are shown in Figure 4.

B. Two-Objective Constrained Problems

Next, we apply G-ASM to two-objective constrained problems: BNH, SRN, TNK, OSY [27], and the welded beam design problem (WB). Table III presents the IGD values of individual frameworks and G-ASM. It can clearly seen that

TABLE III: Computed IGD values for test problems. Best performing framework is marked in bold and other statistically similar frameworks are marked in bold and italic.

Problem	M1-1	M1-2	M2-1	M3-1	M4-1	M5	G-ASM
ZDT1	**0.00090**	0.00555	-	0.00447	-	-	***0.00099***
	-	p=8.1e-5	-	p=8.1e-5	-	-	p=0.5114
ZDT2	***0.00065***	***0.00062***	-	0.00568	-	-	**0.00057**
	p=0.2372	p=0.7928	-	p=8.1e-5	-	-	-
ZDT3	0.06778	**0.00212**	-	0.17123	-	-	***0.00456***
	p=8.1e-5	-	-	p=8.1e-5	-	-	p=0.421
ZDT4	**0.28900**	5.43450	-	***0.29300***	-	-	0.45803
	-	p=8.1e-5	-	p=0.4307	-	-	p=0.0151
ZDT6	0.37058	0.48360	-	***0.24192***	-	-	**0.19038**
	p=0.001	p=8.1e-5	-	p=0.0762	-	-	-
OSY	**0.15323**	***0.18806***	24.57940	6.26550	18.49200	45.18110	***0.16376***
	-	p=0.2643	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	p= 0.4307
TNK	0.01726	**0.00082**	0.04383	0.01180	0.03332	0.03077	***0.00105***
	p=8.1e-5	-	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=0.0569
SRN	**0.13191**	1.00930	4.17160	1.06120	1.20480	1.28450	***0.13434***
	-	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=0.2370
BNH	0.69434	**0.04630**	0.74425	0.23728	0.23923	0.23699	***0.07765***
	p=8.1e-5	-	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=8.1e-5	p=0.652
WB	***0.13794***	***0.23159***	0.55529	***0.16909***	0.88586	0.96166	**0.12819**
	p=0.6457	p= 0.0660	p=8.1e-5	p=0.8438	p=8.1e-5	p=8.1e-5	-
DTLZ2	0.07870	**0.03340**	-	0.05377	-	-	***0.03921***
	p=8.1e-5	-	-	p=8.1e-5	-	-	p=0.1002
C2DTLZ2	0.05130	**0.03355**	-	***0.03493***	-	0.12403	***0.03401***
	p=8.1e-5	-	-	p=0.251	-	p=8.1e-5	p=0.321
CAR	0.43510	0.50119	0.43145	***0.39809***	***0.42223***	0.50061	**0.39060**
	p=0.0041	p=0.0058	p=0.0041	p=0.234	p=0.1149	p=0.0001	-
DTLZ5	0.01960	**0.00948**	-	0.01352	-	-	***0.01123***
	p=8.1e-5	-	-	p=8.1e-5	-	-	p=0.0728
DTLZ4	**0.05840**	0.09024	-	0.20668	-	-	***0.08012***
	-	p=8.1e-5	-	p=8.1e-5	-	-	p=0.205
DTLZ7	0.89316	***0.07664***	-	0.87172	-	-	**0.07261**
	p=8.1e-5	p=0.5994	-	p=8.1e-5	-	-	-
DTLZ2-5	0.21450	**0.03981**	-	0.14401	-	-	***0.04643***
	p=8.1e-5	-	-	p=8.1e-5	-	-	p=0.5109
C2DTLZ2-5	0.17341	**0.03676**	-	0.15388	-	0.29291	***0.04891***
	p= 8.1e-5	-	-	p=8.1e-5	-	p=8.1e-5	p=0.528

in all five problems, G-ASM performs equally well or better than individual frameworks. No individual framework work well on all five constrained problems. Figure 5 shows all final non-dominated archive members for the median performing G-ASM run. Such performances with only a total of 800 high-fidelity SEs are remarkable.

In order to understand the switching mechanism of G-ASM, we calculate the proportion of the times (out of 11 runs) a framework was chosen for the best performing list of frameworks at the end of each epoch. Figure 6 marks this proportion in black and white shading. For example, M1-1 (marked as M11) is always found to be the only best performing framework in all epochs from start to finish for OSY. This is supported by the smallest IGD value of 0.15323 for M1-1 in Table III. By choosing the M1-1 in all epochs by our G-ASM method, it is able to produce statistically similar result on OSY. In WB, G-ASM switches among the six approaches to produce the best IGD value (0.12819), compared to all other individual frameworks. For SRN, BNH and TNK problems, G-ASM finds the frameworks M1-1 and M1-2 mostly useful. A better picture emerges when we plot the switching strategy for a single (median IGD run) in Figure 7 for SRN, BNH and WB. Similar plots for other problems are presented at the supplementary document. G-ASM is able to use a switching of frameworks to produce a better performance on these problems overall. These plots make the power of G-ASM method clear.

C. Three-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 problems (C2DTLZ2

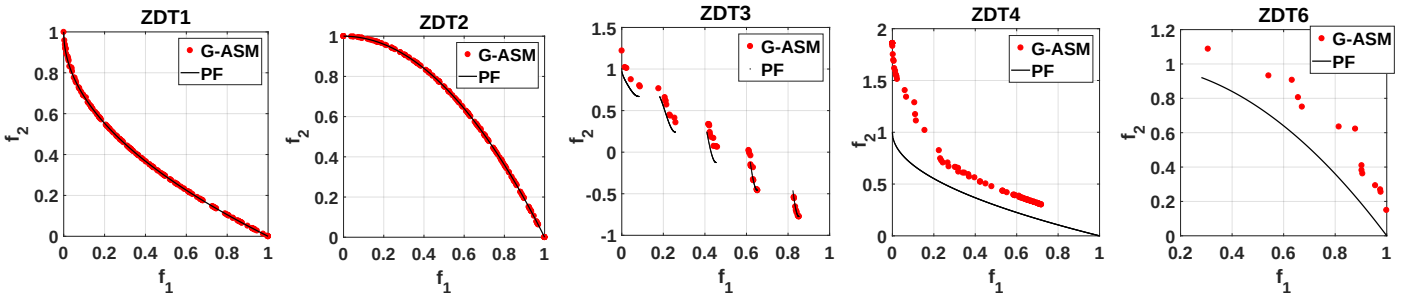

Fig. 4: Obtained non-dominated solutions of the final archive for the median G-ASM run for ZDT problems.

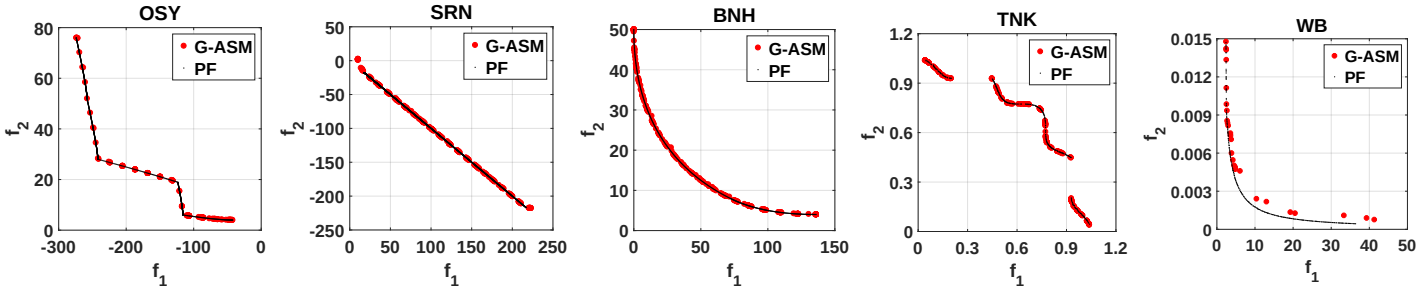

Fig. 5: Obtained non-dominated solutions of the final archive for the median G-ASM run for two-objective constrained problems.

and car side impact problem, CAR). In all six problems, G-ASM performs better or equivalent to individual frameworks. Figure 8 plots the average epoch-wise proportion of usage of frameworks in the G-ASM method in all 11 runs. Patterns of their usage and also the their non-usage of certain frameworks provide interesting information about the frameworks.

D. Five-Objective Constrained and Unconstrained Problems

Next, we apply metamodeling frameworks to five-objective unconstrained DTLZ2 and constrained C2DTLZ2 problems. For both these problems, M1-2 framework (simultaneous optimization) performs the best and our G-ASM, despite having other frameworks as options, choose M1-2 in most runs to produce similar performing IGD (Figure 8 with labels DTLZ2-5 and C2DTLZ2-5). As shown in Table III, none of the other frameworks perform well on these two problems.

E. Summary of Results

The rank of each the six frameworks applied alone and our G-ASM method according to Wilcoxon rank-sum test are shown in Table IV for all 18 problems. It is observed that G-ASM method performs best overall, followed by M1-2. Among the generative frameworks alone, M3-1 (metamodeling an aggregate of objectives and each constraint separately) performs the best.

TABLE IV: Ranking of frameworks summarized. A '1' indicates the best performing framework.

Problem	M1-1	M1-2	M2-1	M3-1	M4-1	M5	G-ASM
ZDT1	1	7	1	4	4	4	1
ZDT2	1	1	1	5	5	5	1
ZDT3	3	1	3	5	5	5	1
ZDT4	1	7	1	1	1	1	6
ZDT6	5	7	5	1	1	1	1
OSY	1	1	6	4	5	7	1
TNK	4	1	7	3	6	5	1
SRN	1	2	6	3	4	5	1
BNH	6	1	7	4	5	3	1
WB	1	1	5	1	6	7	1
DTLZ2	6	1	6	3	3	3	1
C2DTLZ2	5	1	7	1	1	7	1
CAR	5	7	4	1	1	6	1
DTLZ5	6	1	6	3	3	3	1
DTLZ4	1	4	1	5	5	5	1
DTLZ7	6	1	6	3	3	3	1
DTLZ2-5	6	1	6	3	3	3	1
C2DTLZ2-5	5	1	5	3	3	6	1
Average	3.56	2.56	4.50	2.94	3.55	4.50	**1.27**

F. Comparative Study

Finally, we examine the performance of our proposed adaptive switching metamodeling (G-ASM) method by comparing it with three recently-proposed algorithms, namely, MOEA/D-EGO [5], K-RVEA [6], and CSEA [34]. These algorithms are implemented in PlatEMO [37] platform. MOEA/D-EGO and K-RVEA are surrogate-assisted evolutionary multi-objective

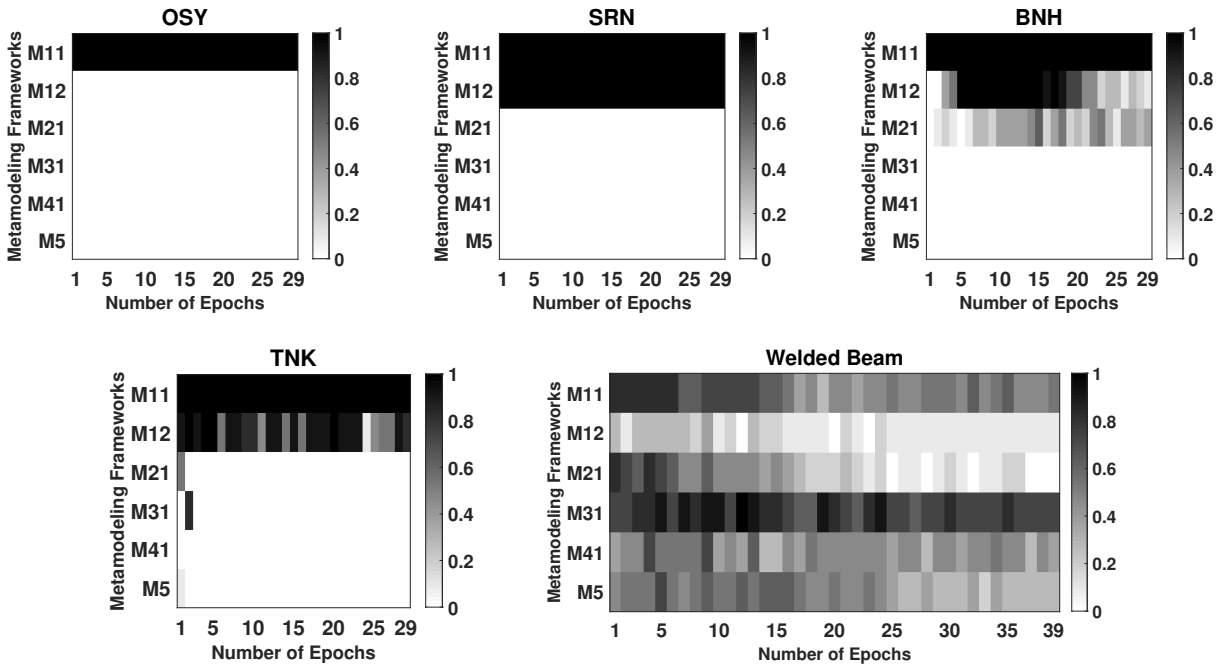

Fig. 6: Proportion of framework usage in 11 runs for two-objective constrained problems.

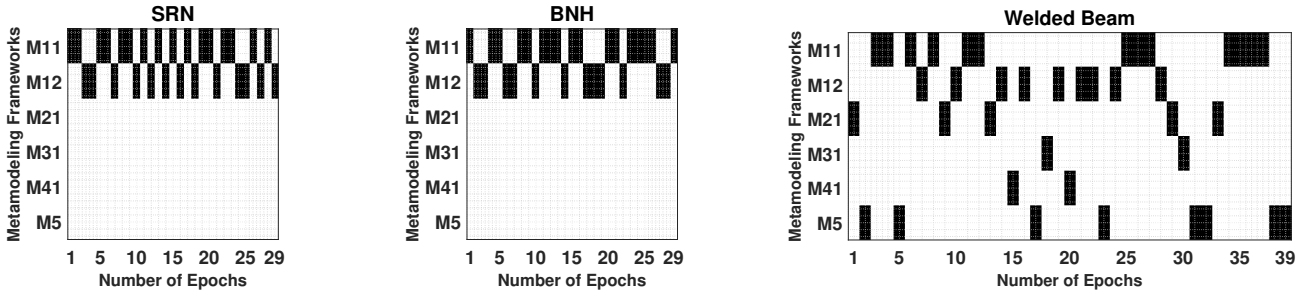

Fig. 7: Switching of frameworks for the median G-ASM run for SRN, BNH and welded beam problems.

optimization methods which use Kriging method to approximate individual objective or an aggregation functions. CSEA is a classification based surrogate-assisted evolutionary algorithm. None of these methods suggest a way to handle constraints. Hence, we restrict our comparative study to ZDT and DTLZ problems only. All methods are compared for the same $\mathrm{SE}_{\max}$.

Table V presents the mean IGD values and marks the best performing method in bold and other methods with $p > 0.05$ using the Wilcoxon rank sum test in bold italics. It is clear that K-RVEA performs the best only on DTLZ4 and DTLZ7 problems. Except on DTLZ4, in all other problems G-ASM performs better or equivalent to K-RVEA. MOEA/D-EGO and CSEA do not perform well on any of the nine problems used in this study.

VI. Conclusions

In the presence of multiple objectives and constraints requiring computationally expensive evaluations, a number of options of metamodeling individual or combined aggregate functions are possible. Recently, authors proposed a taxonomy of 10 metamodeling frameworks for this purpose. In this paper, we have proposed an adaptive switching strategy which chooses a statistically significant framework at the end of each epoch. In this first part of the two-part paper, we restrict the frameworks to be chosen from five generative frameworks and one popular simultaneous framework. Results on two to five-objective constrained and unconstrained test and engineering problems, it has been established that the adaptive switching method performs much better than a single framework. Moreover, since the knowledge of which framework would perform

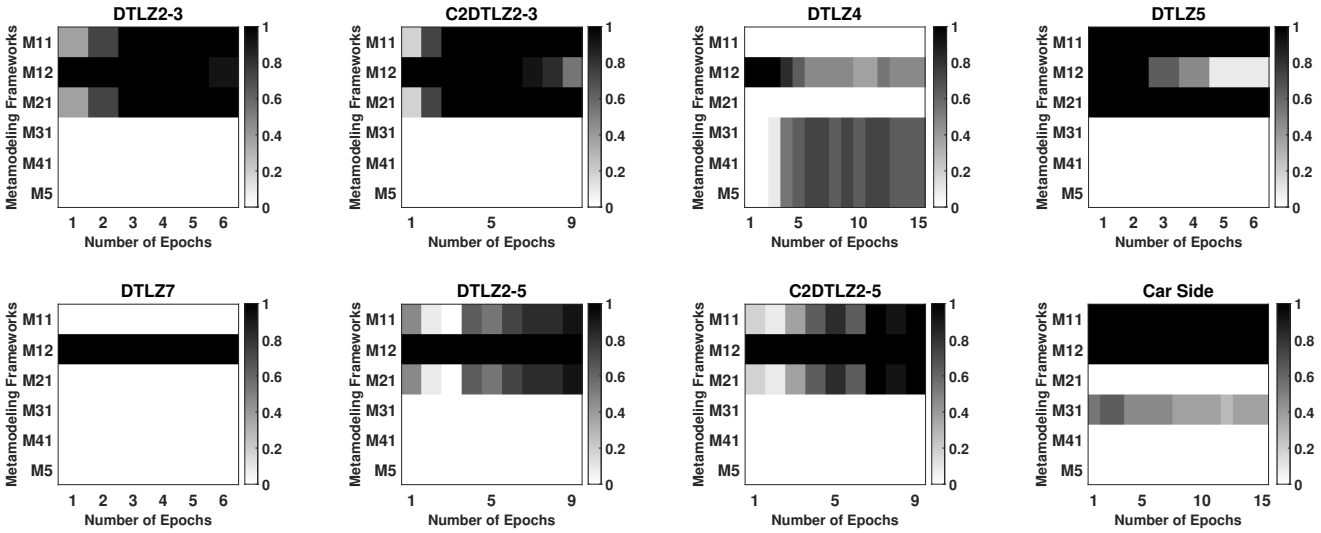

Fig. 8: Proportion of framework usage in 11 runs for three and five-objective problems.

TABLE V: Mean IGD comparison among G-ASM, MOEA/D-EGO, K-RVEA, and CSEA algorithms on unconstrained problems.

Problem	MOEA/D-EGO	K-RVEA	CSEA	G-ASM
ZDT1	0.05611	0.07964	0.95330	**0.00099**
	p=8.1e-5	p=8.1e-5	p=8.1e-5	-
ZDT2	0.04922	0.03395	1.01060	**0.00057**
	p=8.1e-5	p=8.1e-5	p=8.1e-5	-
ZDT3	0.3038	0.02481	0.94841	**0.00456**
	p=8.1e-5	p=8.1e-5	p=8.1e-5	-
ZDT4	73.25920	4.33221	12.71601	**0.45801**
	p=8.1e-5	p=8.1e-5	p=8.1e-5	-
ZDT6	0.51472	0.65462	5.42620	**0.19036**
	p=8.1e-5	p=8.1e-5	p=8.1e-5	-
DTLZ2	0.33170	0.05481	0.11420	**0.03921**
	p=8.1e-5	p=8.1e-5	p=8.1e-5	-
DTLZ4	0.64533	**0.04490**	0.08110	0.08012
	p=8.1e-5	-	p=0.0022	p=0.001
DTLZ5	0.26203	0.01640	0.03081	**0.01123**
	p=8.1e-5	p=0.0262	p=0.0013	-
DTLZ7	5.33220	**0.05310**	0.70520	***0.07261***
	p=8.1e-5	-	p=8.1e-5	p=0.07

the best for a problem is not known a priori, G-ASM method is more appropriate. In order to demonstrate the working of G-ASM method, we have presented an epoch-wise selection of frameworks. In most problems, G-ASM switches between a few frameworks, but in some problems the same framework is used from start to finish. G-ASM has also been found to produce better or equivalent results compared to three newly proposed metamodeling methods on unconstrained problems.

In the second part, we extend the study to include all five simultaneous frameworks first and then consider all 10 generative and simultaneous frameworks for a more detailed investigation.

REFERENCES

[1] A. Cassioli and F. Schoen, "Global optimization of expensive black box problems with a known lower bound," *Journal of Global Optimization*, 2013.

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

[3] W. Ponweiser, T. Wagner, D. Biermann, and M. Vincze, "Multiobjective optimization on a limited budget of evaluations using model-assisted s-metric selection," in *the Parallel Problem Solving from Nature-PPSN*, pp. 784–794, 2008.

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

[5] 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.

[6] 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.

[7] Y. Jin, "A comprehensive survey of fitness approximation in evolutionary computation," *Soft Computing*, vol. 9, no. 1, pp. 3–12, 2005.

[8] S. Z. Martnez and C. A. C. Coello, "Combining surrogate models and local search for dealing with expensive multi-objective optimization problems," in *2013 IEEE Congress on Evolutionary Computation*, pp. 2572–2579, 2013.

[9] I. Loshchilov, M. Schoenauer, and M. Sebag, "A mono surrogate for multiobjective optimization," in *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, pp. 471–478, ACM, 2010.

[10] B. Wilson, D. Cappelleri, T. W. Simpson, and M. Frecker, "Efficient pareto frontier exploration using surrogate approximations," *Optimization and Engineering*, vol. 2, no. 1, pp. 31–50, 2001.

[11] Y. Jin and B. Sendhoff, "A systems approach to evolutionary multiobjective structural optimization and beyond," *IEEE Computational Intelligence Magazine*, vol. 4, no. 3, pp. 62–76, 2009.

[12] Y. Jin, S. Oh, and M. Jeon, "Incremental approximation of nonlinear constraint functions for evolutionary constrained optimization," in *Evolutionary Computation (CEC), 2010 IEEE Congress on*, pp. 1–8, IEEE, 2010.

[13] R. Datta and R. G. Regis, "A surrogate-assisted evolution strategy for constrained multi-objective optimization," *Expert Systems with Applications*, vol. 57, pp. 270–284, 2016.

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

[15] J. Knowles, "Parego: a hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 1, pp. 50–66, 2006.

[16] R. Allmendinger, M. T. Emmerich, J. Hakanen, Y. Jin, and E. Rigoni, "Surrogate-assisted multicriteria optimization: Complexities, prospective solutions, and business case," *Journal of Multi-Criteria Decision Analysis*, vol. 24, no. 1-2, pp. 5–24, 2017.

[17] P. C. Roy and K. Deb, "High dimensional model representation for solving expensive multi-objective optimization problems," in *CEC'2016*, 2016.

[18] 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.

[19] 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.

[20] K. Deb and R. Datta, "Hybrid evolutionary multi-objective optimization and analysis of machining operations," *Engineering Optimization*, 2012.

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

[22] K. Miettinen and M. M. Mäkelä, "On scalarizing functions in multiobjective optimization," *OR spectrum*, vol. 24, no. 2, pp. 193–213, 2002.

[23] M. Pescador-Rojas, R. H. Gómez, E. Montero, N. Rojas-Morales, M.-C. Riff, and C. A. C. Coello, "An overview of weighted and unconstrained scalarizing functions," in *International Conference on Evolutionary Multi-Criterion Optimization*, pp. 499–513, Springer, 2017.

[24] Y.-S. Ong, Z. Zhou, and D. Lim, "Curse and blessing of uncertainty in evolutionary algorithm using approximation," in *2006 IEEE International Conference on Evolutionary Computation*, pp. 2928–2935, 2006.

[25] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black-box functions," *J. of Global Optimization*, 1998.

[26] K. Deb, "An efficient constraint handling method for genetic algorithms," *Computer methods in applied mechanics and engineering*, 2000.

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

[28] A. P. Wierzbicki, "The use of reference objectives in multiobjective optimization," in *Multiple criteria decision making theory and application*, pp. 468–486, Springer, 1980.

[29] I. Das and J. E. Dennis, "Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems," *SIAM J. on Optimization*, vol. 8, no. 3, 1998.

[30] 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.

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

[32] 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.

[33] A. A. Rahat, R. M. Everson, and J. E. Fieldsend, "Alternative infill strategies for expensive multi-objective optimisation," in *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 873–880, ACM, 2017.

[34] 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.

[35] M. N. Le, Y. S. Ong, S. Menzel, C.-W. Seah, and B. Sendhoff, "Multi co-objective evolutionary optimization: Cross surrogate augmentation for computationally expensive problems," in *Evolutionary Computation (CEC), 2012 IEEE Congress on*, pp. 1–8, IEEE, 2012.

[36] 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.

[37] Y. Tian, R. Cheng, X. Zhang, and Y. Jin, "platemo: A matlab," *IEEE Computational Intelligence Magazine*.