Towards Efficient Multiobjective Optimization: Multiobjective Statistical Criterions

Ivo Couckuyt, Dirk Deschrijver, and Tom Dhaene

***Abstract*—The use of Surrogate Based Optimization (SBO) is widely spread in engineering design to reduce the number of computational expensive simulations. However, "real-world" problems often consist of multiple, conflicting objectives leading to a set of equivalent solutions (the Pareto front). The objectives are often aggregated into a single cost function to reduce the computational cost, though a better approach is to use multiobjective optimization methods to directly identify a set of Pareto-optimal solutions, which can be used by the designer to make more efficient design decisions (instead of making those decisions upfront). Most of the work in multiobjective optimization is focused on MultiObjective Evolutionary Algorithms (MOEAs). While MOEAs are well-suited to handle large, intractable design spaces, they typically require thousands of expensive simulations, which is prohibitively expensive for the problems under study. Therefore, the use of surrogate models in multiobjective optimization, denoted as MultiObjective Surrogate-Based Optimization (MOSBO), may prove to be even more worthwhile than SBO methods to expedite the optimization process. In this paper, the authors propose the Efficient Multiobjective Optimization (EMO) algorithm which uses Kriging models and multiobjective versions of the expected improvement and probability of improvement criterions to identify the Pareto front with a minimal number of expensive simulations. The EMO algorithm is applied on multiple standard benchmark problems and compared against the well-known NSGA-II and SPEA2 multiobjective optimization methods with promising results.**

***Index Terms*—multiobjective optimization, Kriging, expected improvement, probability of improvement**

I. Introduction

This paper is concerned with efficiently solving complex, computational expensive design problems using surrogate modeling techniques [1]. Surrogate models, also known as metamodels, are cheap approximation models for computational expensive (black-box) simulations. Surrogate modeling techniques are well-suited to handle, for example, expensive mechanical or electrical finite element simulations, or computational fluid dynamic simulations. In particular, this paper deals mainly with deterministic computer codes, as opposed to non-deterministic (stochastic) problems.

Depending on the construction and usage of surrogate models, several modeling flavors can be distinguished. Surrogate models can be built upfront to approximate the simulation code accurately over the entire input (design) space and, hence, can afterwards be used to replace the expensive code for design, analysis and optimization purposes. On the other hand, the construction of surrogate models can also be integrated in the optimization process. Usually, the latter case, known as Surrogate Based Optimization (SBO), generates surrogate models on the fly that are only accurate in certain regions of the input space, e.g., around potentially optimal regions.

Developing the most efficient surrogate models is an entire research domain in itself. In order to come to an acceptable model, numerous problems and design choices need to be overcome *(what data collection strategy to use*, *which variables are relevant*, *how to integrate domain knowledge, etc.*). Other aspects of surrogate modeling include choosing the right type of approximation model for the problem at hand, a tuning strategy for the surrogate model parameters (=hyperparameters), and a performance measure to assess the accuracy of the surrogate model [2].

The focus of this work is the global SBO method based on the Probability of Improvement (PoI) and Expected Improvement (EI), popularized by Jones et al. [3]. These "*statistical criterions*" guide the selection of new data points in such a way that the objective function is optimized, while minimizing the number of expensive simulations. The advantage of EI and PoI is that, besides the prediction mean of the surrogate model, the prediction variance (uncertainty) is taken into account as well, providing a balance between exploration[1] and exploitation[2]. Most often EI or PoI is used in conjunction with the Kriging surrogate model (Gaussian Processes) [4], but other surrogate models are also possible, such as Radial Basis Functions (RBF), Support Vector Regression (SVR) [5], etc.

The single-objective SBO problem is well described in literature, however, most (if not all) "real-world" problems actually consists of multiple, conflicting objectives leading to a set of Pareto-optimal solutions. A multiobjective optimization method can optimize the different objective functions simultaneously, and try to find the Pareto front in just a single run. Examples of such methods are primarily the Multiobjective Evolutionary Algorithms (MOEAs), e.g., the "Non-dominated Sorting Genetic Algorithm II" (NSGA-II; [6]), the "Strength Pareto Evolutionary Algorithm 2" (SPEA2; [7]) and the "$\mathcal{S}$-Metric Selection Evolutionary MultiObjective Algorithm" (SMS-EMOA; [8]).

Unfortunately, MOEAs typically require a massive amount of function evaluations, which is infeasible for computational expensive simulators. Hence, it is vital to economize on the number of function evaluations, e.g., by using surrogate models. MultiObjective Surrogate-based Optimization (MOSBO) methods only appeared quite recently in literature. Most work is focused on integrating surrogate models in MOEAs [9]. Gaspar et al. use neural networks to either approximate the

[1]Improving the overall accuracy of the surrogate model (space-filling).
[2]Enhancing the accuracy of the surrogate model solely in the region of the (current) optimum.

fitness function or as a local approximation technique to generate search points more efficiently [10]. Voutchkov et al. [11] apply the NSGA-II algorithm to Kriging models instead of the expensive simulator. For an overview of available techniques and approaches, the reader is referred to [12], [13].

While the PoI and EI approach is well-developed and used for single-objective SBO, its use in MOSBO is not well spread. Single-objective versions of EI and PoI are utilized by Knowles et al. [14], [15] to solve MOSBO problems. This approach, known as ParEGO, uses Kriging and EI to optimize a weighted sum of objective functions. By randomizing the weights every iteration several solutions along the Pareto front can be identified. More recently, Keane [16] proposed multiobjective versions of PoI and EI with promising results. Similarly to a weighted sum, the multiobjective versions of EI and PoI aggregate information from the surrogate models into a single cost function, balancing between exploration[1] and exploitation[3]. Unfortunately, only formulae for two objective functions are given by Keane as the statistical criterions become rather cumbersome and complex for a higher number of objective functions.

The contribution of this paper is the Efficient Multiobjective Optimization (EMO) algorithm which generalizes the multi-objective versions of the PoI and EI criterion to an arbitrary number of objective functions. In fact, the problem is quite similar to calculating the hypervolume (a Pareto set quality estimator) [17] as will be shown in this paper.

In section II the Kriging surrogate model is briefly discussed. An overview of the EMO algorithm is given in section III-A. General expressions for PoI and EI are given in section III-B. Subsequently, a fundamental part needed for the calculation of the statistical criterions, the branch-and-bound procedure, is discussed in section III-C. Afterwards, the EMO algorithm is applied to functions from the DTLZ benchmark suite [18] in section IV. Lastly, conclusions and future work are described in section V.

II. Kriging

Kriging is a popular surrogate model to approximate deterministic noise-free data, and has proven to be very useful for tasks such as optimization [3], design space exploration, visualization, prototyping, and sensitivity analysis [1].

A thorough mathematically treatment of Kriging is given in [19], [20]. Basically, Kriging is a two-step process: first a regression function $h(\mathbf{x})$ is constructed based on the data, and, subsequently, a Gaussian process Z is constructed through the residuals.

$$Y(\mathbf{x}) = h(\mathbf{x}) + Z(\mathbf{x}), \tag{1}$$

where $h(\mathbf{x})$ is a regression function and Z is a Gaussian process with mean 0, variance σ^2 and a correlation matrix Ψ.

[3]Improving or augmenting the Pareto front.

Consider a set of n samples, $(\mathbf{x}_1, \ldots, \mathbf{x}_n)'$ in d dimensions (see Equation 2) and associated function values, $\mathbf{y} = (y_1, \ldots, y_n)'$, where $(\cdot)'$ is the transpose of a vector or matrix.

$$X = \begin{pmatrix} \mathbf{x}_1, \ldots, \mathbf{x}_n \end{pmatrix}' = \begin{pmatrix} x_{1,1} & \ldots & x_{1,d} \\ \vdots & \ddots & \vdots \\ x_{n,1} & \ldots & x_{n,d} \end{pmatrix} \tag{2}$$

Essentially, the regression part is encoded in the $n \times p$ model matrix F using basis functions $b_i(\mathbf{x})$ for $i = 1 \ldots p$,

$$F = \begin{pmatrix} b_1(\mathbf{x_1}) & b_2(\mathbf{x_1}) & \cdots & b_p(\mathbf{x_1}) \\ \vdots & \vdots & \vdots & \vdots \\ b_1(\mathbf{x_n}) & b_2(\mathbf{x_n}) & \cdots & b_p(\mathbf{x_n}) \end{pmatrix},$$

while the stochastic process is mostly defined by the $n \times n$ correlation matrix Ψ,

$$\Psi = \begin{pmatrix} \psi(\mathbf{x}_1, \mathbf{x}_1) & \ldots & \psi(\mathbf{x}_1, \mathbf{x}_n) \\ \vdots & \ddots & \vdots \\ \psi(\mathbf{x}_n, \mathbf{x}_1) & \ldots & \psi(\mathbf{x}_n, \mathbf{x}_n) \end{pmatrix},$$

where $\psi(\cdot, \cdot)$ is the correlation function. $\psi(\cdot, \cdot)$ is parametrized by a set of hyperparameters θ, which are identified by Maximum Likelihood Estimation (MLE) (though other approaches are possible). Subsequently, the prediction mean and prediction variance of Kriging are derived, respectively, as,

$$\mu(\mathbf{x}) = M\alpha + r(\mathbf{x}) \cdot \Psi^{-1} \cdot (\mathbf{y} - F\alpha), \tag{3}$$

$$s^2(\mathbf{x}) = \sigma^2 \left(1 - r(\mathbf{x})\Psi^{-1} r(\mathbf{x})' + \frac{(1 - F'\Psi^{-1} r(\mathbf{x})')}{F'\Psi^{-1}F} \right), \tag{4}$$

where $M = \begin{pmatrix} b_1(\mathbf{x}) & b_2(\mathbf{x}) & \ldots & b_p(\mathbf{x}) \end{pmatrix}$ is the model matrix of the predicting point $\mathbf{x}$, α is a $p \times 1$ vector denoting the coefficients of the regression function, determined by Generalized Least Squares (GLS), and $r(\mathbf{x})$ is an $1 \times n$ vector of correlations between the point $\mathbf{x}$ and the samples X.

III. Efficient Multiobjective Optimization (EMO)

A. Overview

A flow chart of the EMO algorithm is shown in Figure 1. First an initial set of points X is generated and evaluated on the expensive objective functions $f_i(\mathbf{x})$, for $i = 1 \ldots m$. Each objective function $f_i(\mathbf{x})$ is then approximated by a Kriging model. Based on the Kriging models useful statistical criterions can be constructed that help in identifying Pareto-optimal solutions. In particular, a new point is selected by optimizing the statistical criterion. Finally, the new point is evaluated on the expensive objective functions $f_i(\mathbf{x})$, the Kriging models are updated with this new information and this process is repeated in an iterative fashion until some stopping criterion is met.

Of particular interest are the Probability of Improvement (PoI) and Expected Improvement (EI) statistical criterions,

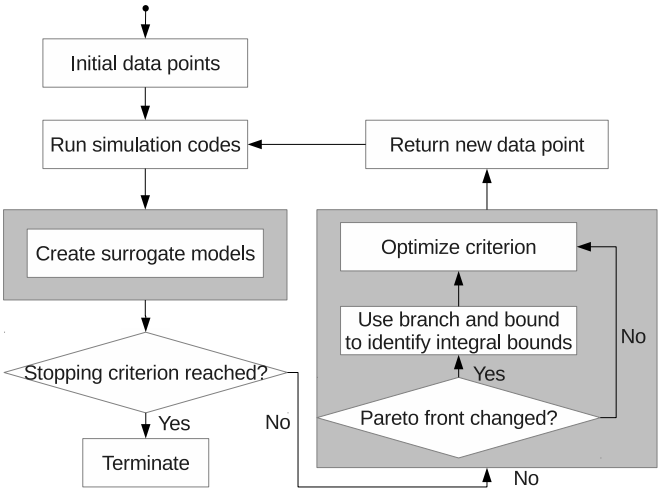


Figure 1: Flow chart of the Efficient Multiobjective Optimization (EMO) algorithm.

which are widely used for single-objective optimization [21], [22]. Hence, it may be useful to extend the concept of the PoI and EI to multiobjective optimization. Multiobjective versions of the PoI and EI have been suggested by Keane [16] for two objective functions, and later extended using the concept of Level of Improvement (LoI; [23]). The LoI generates several variants of PoI and EI by considering how many points in the Pareto set are dominated by a new input point $\mathbf{x}$, see Figure 2a. In this work, statistical criterions are defined based on a minimal LoI value. For instance, the PoI with a minimum LoI of three denotes the probability that a new point dominates three or more points of the Pareto set. Logically, a LoI of zero or greater then leads to the probability that a new point extends or dominates the Pareto set. The statistical functions are defined for an arbitrary number of objective functions in section III-B.

In order to evaluate these statistical criterions, a branch-and-bound procedure is presented in section III-C that decomposes the overall objective space into a set of smaller hyperrectangles. The upper and lower bounds of the hyperrectangles that dominate or augment the Pareto front (depending on the LoI) are used to evaluate the statistical criteria. This procedure can be computationally expensive, but fortunately it has to be done at most once per sampling iteration. Regardless, note that the applicability of the EMO algorithm to large scale problems is also limited by the Kriging model for > 20 dimensions and > 10000 points.

B. Statistical criterions

The output of all the Kriging models can be considered as mutually independent Gaussian random variables $Y_i(\mathbf{x})$,

$$Y_i(\mathbf{x}) \sim \mathcal{N}(\mu_i(\mathbf{x}), s_i^2(\mathbf{x})) \; for \; i = 1 \dots m. \tag{5}$$

The associated probability density function and cumulative distribution function of $Y_i(\mathbf{x})$ are compactly denoted as,

$$\phi_i[y_i] \triangleq \phi_i[y_i; \mu_i(\mathbf{x}), s_i^2(\mathbf{x})], \tag{6}$$

$$\Phi_i[y_i] \triangleq \Phi_i[y_i; \mu_i(\mathbf{x}), s_i^2(\mathbf{x})]. \tag{7}$$

Given an initial set of n points X as in (2), a Pareto set $\mathcal{P}$ can be constructed that comprises $v \leq n$ Pareto-optimal (non-dominated) solutions,

$$\mathcal{P} = \{\mathbf{f}(\mathbf{x}_1^*), \dots, \mathbf{f}(\mathbf{x}_v^*)\}. \tag{8}$$

Each solution $\mathbf{f}(\mathbf{x}_i^*)$ is essentially a vector that contains the objective function values for an associated input point $\mathbf{x}_i^* \in X$, for $i = 1 \dots v$,

$$\mathbf{f}(\mathbf{x}_i^*) = (f_1(\mathbf{x}_i^*), \dots, f_m(\mathbf{x}_i^*)). \tag{9}$$

As illustrated in Figure 1, the algorithm needs to select a new point $\mathbf{x}$ whose solution $\mathbf{f}(\mathbf{x})$ extends or dominates a minimum number of solutions in the Pareto set $\mathcal{P}$ according to a given LoI. The probability that a new input point $\mathbf{x}$ yields this improvement is denoted by the PoI $P[I]$,

$$P[I] = \sum_{j=1}^{k} P\left(\bigcup_{i=1}^{m} (l_i^j \leq Y_i(\mathbf{x}) \leq u_i^j)\right), \tag{10}$$

where $[\mathbf{l}^j, \mathbf{u}^\mathbf{j}]$ are non-overlapping hyperrectangles in the objective space that dominate a minimum of LoI points of the Pareto set $\mathcal{P}$. Note that $P[I]$ represents the probability that the outcome of all $Y_i(\mathbf{x})$ is located inside one of the non-overlapping hyperrectangles that dominate the Pareto set $\mathcal{P}$, depending on the given LoI (see, e.g., grey hyperrectangles in Figure (2a)). These hyperrectangles are non-overlapping, and have a certain lower and upper bound $[\mathbf{l}^j, \mathbf{u}^j]$ that will be computed in section (III-C).

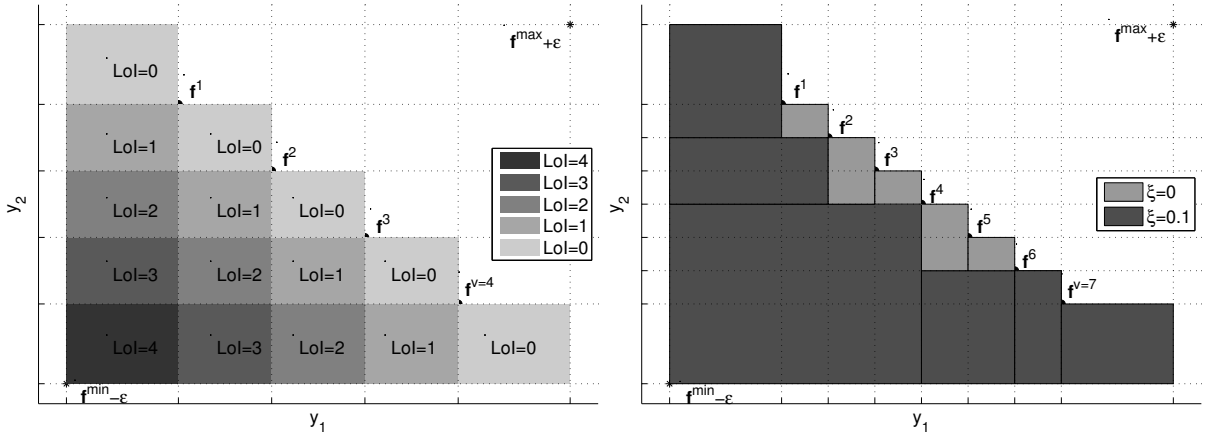
The probability that the outcome of all $Y_i(\mathbf{x})$ is located inside a single hyperrectangle of the sample space with lower and upper bound $[l_i^j, u_i^j]$ is given as,

$$\begin{aligned} & P\left(\bigcup_{i=1}^{m} l_i^j \leq Y_i(\mathbf{x}) \leq u_i^j\right) \\ &= \int_{l_1^j}^{u_1^j} \dots \int_{l_m^j}^{u_m^j} \left(\prod_{i=1}^{m} \phi_i[y_i]\right) dy_m, \dots, dy_1 \\ &= \prod_{i=1}^{m} (\Phi_i[u_i^j] - \Phi_i[l_i^j]). \end{aligned} \tag{11}$$

While the PoI criterion is already quite useful and insensitive to the scaling of the objective functions, it does not, necessarily, encourage the generation of an uniform Pareto set. The EI $E[I]$ quantifies the amount of improvement and, thus, prefers solutions that are lying farther from existing members of the Pareto set. $E[I]$ for an input vector $\mathbf{x}$ is defined as,

$$E[I] = P[I] \cdot \sqrt{\sum_{i=1}^{m} \alpha_i (\hat{\mathbf{y}}_i(\mathbf{x}) - \mathbf{f}_i^c)}, \tag{12}$$

where $\hat{\mathbf{y}}(\mathbf{x})$ denotes the centroid of the $P[I]$ integral,



(a) (b)

Figure 2: Illustration of a Pareto set of two objective functions. The dots represent the Pareto points $\mathbf{f}^i$, for $i = 1 \dots v$, while $\mathbf{f}^{min}$ and $\mathbf{f}^{max}$ denote the ideal and anti-ideal point, respectively. a) The shaded regions (hyperrectangles) denote the Level of Improvement (LoI), namely, the number of Pareto points that are dominated by that region. b) The shaded regions (light and dark) correspond to the sets of hyperrectangles identified by the branch-and-bound procedure using $LoI = 0$ (dominated or augmented by the Pareto set) and $\xi = 0$. As the number of Pareto points increase, the regions close to the Pareto points will likely be smaller and correspond to the small improvements over the Pareto set. Hence, a much faster, approximated, version of the EMO algorithm can be used by ignoring the smaller regions, for example, the light shaded regions are not considered anymore when using $\xi = 0.1$.

$$\hat{y}_i(\mathbf{x}) = \sum_{j=1}^{k} \hat{y}_i(\mathbf{x}; \mathbf{l}^j, \mathbf{b}^j). \tag{13}$$

Note that a predefined weight vector α allows a designer to scale the objective functions in advance, although techniques can be used to automatically identify them. The weighted norm in (12) represents the Euclidean distance between the centroid $\hat{\mathbf{y}}(\mathbf{x})$ and the solution in $\mathcal{P}$ that is located closest to the centroid, i.e., $\mathbf{f}^c$,

$$\mathbf{f}^c = \underset{\mathbf{f}^c \in \mathcal{P}}{\operatorname{argmin}} \sqrt{\sum_{i=1}^{m} \alpha_i (\hat{\mathbf{y}}_i(\mathbf{x}) - \mathbf{f}_i^c)}. \tag{14}$$

The centroid $\hat{\mathbf{y}}$ over arbitrary integral bounds $[\mathbf{l}, \mathbf{b}]$ is then defined by,

$$\begin{aligned} &\hat{y}_i(\mathbf{x}; \mathbf{l}, \mathbf{b}) = \\ &\int_{l_1}^{u_1} \dots \int_{l_m}^{u_m} \phi_1[y_1] \dots y_i \phi[y_i] \dots \phi_m[y_m] \, dy_m \dots dy_1 \big/ P[I] = \\ &\prod_{j=1}^{i-1} (\Phi_j[u_j] - \Phi_j[l_j]) \times \prod_{j=i+1}^{m} (\Phi_j[u_j] - \Phi_j[l_j]) \times \\ &\left(\mu_i(\mathbf{x})\Phi_i[u_i] - s_i^2(\mathbf{x})\phi_i[u_i] - \mu_i(\mathbf{x})\Phi_i[l_i] + s_i^2(\mathbf{x})\phi_i[l_i]\right) \big/ P[I] \end{aligned} \tag{15}$$

C. Branch-and-bound procedure

To calculate the statistical functions defined in the previous section, the integral bound of the region that improves the Pareto set for a certain LoI needs to be identified. As this region is non-rectangular and often irregularly shaped, especially for a higher number of objective functions, the integral over that region is decomposed into a sum of k integrals over smaller, connected (but non-overlapping) hyperrectangles, see (10) and (13). While these integral bounds can be calculated analytically upfront, as done in [16], this becomes rather prohibitively complex and cumbersome for a higher number of objective functions (> 2).

Instead, the authors propose to identify the integral bounds using a computer algorithm. First, a pseudo Pareto set $\bar{\mathcal{P}}$ is constructed by augmenting the Pareto set $\mathcal{P}$ with two additional points, namely, $\mathbf{f}^{min} - \epsilon$ and $\mathbf{f}^{max} + \epsilon$ for some $\epsilon > 0$, representing $(-\infty, \dots, -\infty)$ and $(\infty, \dots, \infty)$, respectively. Where $\mathbf{f}^{min}$ is the ideal point and $\mathbf{f}^{max}$ is the anti-ideal point of the Pareto set $\mathcal{P}$. The grid defined by the coordinates of the points in $\bar{\mathcal{P}}$ partitions the objective space in $(v+1)^m$ hyperrectangles, see Figure 2a. All points inside a single hyperrectangle dominate (or augment) the exact same number of Pareto points from the Pareto set $\mathcal{P}$. Hence, to decide whether a hyperrectangle dominates (or augments) the Pareto set it suffices to only test one point of the hyperrectangle, e.g., the test point $\mathbf{f}^{test}$ in the middle of the hyperrectangle.

The idea is to identify the smallest set of hyperrectangles, and the associated integral bounds $[\mathbf{l}^j, \mathbf{u}^j]$, that dominate (or augment) the Pareto set for a certain LoI. To that end, a binary partitioning strategy is employed that, starting from the initial set of $(v+1)^m$ hyperrectangles, recursively partitions the objective space into smaller sets of hyperrectangles until all hyperrectangles have been safely accepted or rejected based on a boolean condition function $g(\mathbf{f}^{test}; \mathcal{P}, LoI)$, see Figure 2a. The condition function $g(\mathbf{f}^{test}; \mathcal{P}, LoI)$ evaluates whether a test point $\mathbf{f}^{test}$ dominates at least LoI Pareto points of $\mathcal{P}$. Consequently, a set of hyperrectangles can be readily

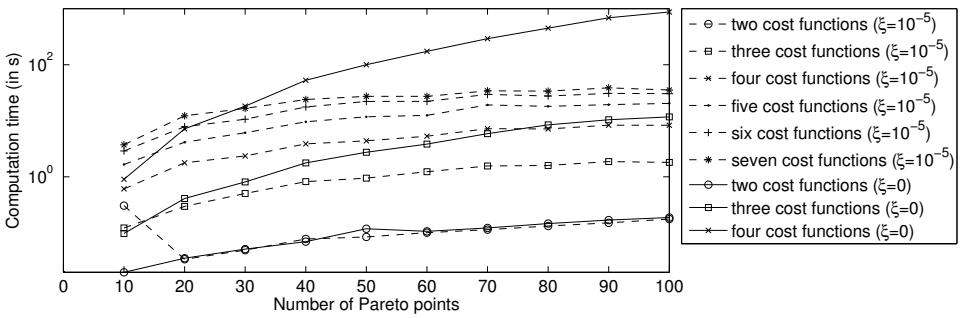


Figure 3: Computation time of the integral bounds versus the number of Pareto points, for a different number of objective functions. The exact EMO algorithm ($\xi = 0$) can be applied to four or five objective functions, depending on the density of the Pareto set. By using a slightly approximated version of the EMO algorithm ($\xi = 10^{-5}$) a higher number of objective functions is possible..

accepted if the hyperrectangle lying closest to the anti-ideal point satisfies the condition function. Due to the nature of the condition function implicitly all the other hyperrectangles in the set will also satisfy the condition function. Similarly, a set of hyperrectangles can be immediately rejected if the hyperrectangle closest to the ideal point does not satisfy the condition function. In all other cases, i.e., the set of hyperrectangles can neither be accepted nor rejected, the set of hyperrectangles is divided along its longest edge (= binary partitioning, though other divide steps can be used) according to its enclosing hyperrectangle and the process is repeated on both newly created sets of hyperrectangles.

After $k << (v+1)^m$ sets of hyperrectangles (=integral bounds) have been identified the actual PoI and EI statistical criterions can be evaluated using Equations (10) and (13). Note, that for evaluating the criterions the points $\mathbf{f}^{min} - \epsilon$ and $\mathbf{f}^{max} + \epsilon$ are replaced by $(-\infty, \ldots, -\infty)$ and $(\infty, \ldots, \infty)$, respectively.

The overall computation time may be a problem, however, it is crucial to point out that the computational expensive part of the algorithm, i.e., identifying the integral bounds, is completely separated from the calculation of the final PoI and EI criterions. Another key benefit is that one of the more interesting Pareto set quality estimators, i.e., the hypervolume indicator (see section III-D), is obtained with no additional cost. Furthermore, the integral bounds need to be calculated just once every iteration, and this only when the Pareto set actually changed (improved or augmented) by the new point. A last remark is that during the optimization process the number of points on the Pareto set may actually decrease as a new point can dominate former Pareto points. Only when the objective space has been explored sufficiently the EMO algorithm will converge to a more dense (and uniform) Pareto set.

Nonetheless, it is almost infeasible to calculate the statistical criterions for a high number of objective functions (≥ 5). To that end, the algorithm can be modified to reject a set of hyperrectangles early when its hypervolume becomes smaller than some threshold ξ, namely, a fraction of the total hypervolume of the hyperrectangle enclosing the whole Pareto set. The time savings are significant and allow the algorithm to be applied to higher dimensional problems with some loss of accuracy. The effect of neglecting the small hyperrectangles is that as the intermediate Pareto set becomes denser, the smaller regions will often be located closer to the Pareto set. Hence, after a certain point in the optimization process small improvements to the Pareto set (whether by improving it or augmenting the front) will not be considered anymore by the EI and PoI criterions, see Figure 2b. Of course, at that point, the intermediate Pareto set might be dense enough for the problem at hand. Still, a possible solution to alleviate this issue is to use an initial rough approximation ($\xi > 0$) of the integral bounds, and update the integral bounds exact ($\xi = 0$) every iteration instead of recalculating them, taking full advantage of the sequential design (adaptive sampling) step in the surrogate modeling process.

For all of the above reasons and considering the expensive nature of the optimization problems, the EMO algorithm can be applied to problems with four or five objective functions with no accuracy loss (depending on the density of the Pareto solutions), while the approximated version of the multiobjective EI and PoI can be used for a higher number of objective functions. A plot with the practical computation time of the integral bounds is shown in Figure 3, applying the EMO algorithm with $\xi = 0$ and $\xi = 10^{-5}$ to sets of Pareto points randomly drawn from the first quadrant of a unit sphere. Note that the computation time may actually change with different shapes of the Pareto set.

D. Hypervolume

There exists several quality estimators for multiobjective optimization. Arguably, one of the better quality estimators is the hypervolume [17], as it has the desired property of strict Pareto compliance, namely, a Pareto set A is considered better than Pareto set B if and only if A dominates B. In essence, the hypervolume indicator computes the size (hypervolume or Lebesgue integral) between the (intermediate) Pareto set and some reference point $\mathbf{r}$. This reference point $\mathbf{r}$ needs to be dominated by all points of the (intermediate) Pareto set (at all times if calculated each iteration during the optimization process). Larger values of the hypervolume indicate a better Pareto set. Note that the dominated hypervolume (or $\mathcal{S}$-metric)

Table I: Summary of the DTLZ benchmark functions.

Function	d	m	reference point r
DTLZ2	6 inputs	3 objectives	$(2.5, 2.5, 2.5)$
DTLZ7	6 inputs	4 objectives	$(1, 1, 1, 50)$
DTLZ5	6 inputs	6 objectives	$(2.5, 2.5, 2.5, 2.5, 2.5, 2.5)$

has also been used in MOEAs to drive the multiobjective optimization process [8].

Unfortunately, computing the hypervolume does not scale very well with the number of objective functions m and the number of Pareto points v. Though exact algorithms [24] as well as approximations [14] and alternative versions of the hypervolume problem, e.g., finding the Pareto point(s) that contributes least to the hypervolume [25], have been suggested in literature. The EMO algorithm is quite similar to the hypervolume problem. Whereas the hypervolume can be seen as the Lebesgue integral of the hyperrectangle that dominates the Pareto set, the EMO algorithm needs to calculate an integral over that same hyperrectangle, i.e., if $LoI = 0$, for a different integrand. Moreover, the statistical criterions are functions with respect to a point $\mathbf{x}$. Hence, the EMO algorithm identifies the integral bounds in a pre-processing step so that afterwards the statistical criterions can be evaluated multiple times (with different $\mathbf{x}$). Likewise, the EMO algorithm can be leveraged to calculate the hypervolume as follows.

Assuming the bounds $[\mathbf{l}^j, \mathbf{u}^j]$ for $j = 1 \dots k$ have been calculated for $LoI = 0$, then the hypervolume $H(\mathbf{r})$ can be expressed as,

$$H(\mathbf{r}) = \prod_{j=1}^{m} (r_j - f_j^{min}) - \sum_{j=1}^{k} \left(\int_{l_1^j}^{u_1^j} \dots \int_{l_m^j}^{u_m^j} 1 \, dy_m \dots dy_1 \right), \tag{16}$$

where $\mathbf{f}^{min} - \epsilon$ and $\mathbf{f}^{max} + \epsilon$ from the branch-and-bound procedure are here replaced by just $\mathbf{f}^{min}$ and $\mathbf{r}$, respectively. Note that when the EMO algorithm is configured with $\xi > 0$ the calculated hypervolume is an overestimation. Lastly, the hypervolume is a relative error as the reference point $\mathbf{r}$ must be chosen identical when comparing Pareto sets generated by different multiobjective optimization algorithms.

IV. Examples

A. Introduction

A good set of configurable multiobjective benchmark problems has been proposed by Deb et al. [18], of which three benchmark functions are chosen and adapted slightly to benchmark the EMO algorithm. A summary of the selected benchmark functions is found in Table I. For a complete description of the benchmark functions the authors refer to [18].

All benchmark functions are configured to have six input parameters. Specifically, the first example is the DTLZ2 function with three objective functions where the Pareto front is the first quadrant of an unit sphere centered on the origin. The second example is the DTLZ7 function with four objective functions which has $2^{m-1} = 2^{4-1} = 8$ disconnected Pareto-optimal regions in the objective space. The last example, the DTLZ5 function configured to have six objective functions, is similar to DTLZ2 except that the Pareto front is just one slice of the unit hypersphere, i.e., the Pareto front is a (densely populated) curve in a $m = 6$ dimensional objective space.

B. Experimental setup

An initial set of 65 samples is generated by a near-optimal maximin Latin Hypercube Design (LHD; [26]). Subsequently, the EI criterion is optimized each iteration to select the next point to evaluate. In particular, the condition function is configured with $LoI = 0$ is, namely, $g(\mathbf{f}^{test}; \mathcal{P}, LoI = 0) = \exists \mathbf{f} \in \mathcal{P} : f_1^{test} < f_1 \vee \dots \vee f_m^{test} < f_m$. Furthermore, the optional weight vector α is set to $\mathbf{1}$ for all benchmark functions, except for DTLZ7 where $\alpha = (1, 1, 1, 0.02)$. Two EMO runs are applied to each benchmark function, the first run is configured with $\xi = 0$ (exact version of the EI) and the other run has $\xi = 10^{-5}$ (slightly approximated version of the EI). Except for the DTLZ5 benchmark function, where only the second EMO run is applied, i.e., $\xi = 10^{-5}$.

The EI criterion is optimized using a combination of Monte Carlo sampling and a local search. Specifically, $100 \times n$ Monte Carlo candidate points are generated and evaluated on the EI criterion. Subsequently, the best Monte Carlo candidate is further refined using Matlab's fmincon optimizer.

Lastly, the Kriging models are configured using the Matérn correlation function [27] with $\nu = \frac{3}{2}$ while the hyperparameters (=length scales) are optimized using SQPLab [28] (http://www-rocq.inria.fr/~gilbert/modulopt/optimization-routines/sqplab/sqplab.html), utilizing likelihood derivative information. The EMO algorithm halts when the sample budget is met, namely, 250 samples.

The EMO algorithm runs are compared against the NSGA-II and SPEA2 evolutionary algorithms with a varying population size and maximum number of generations. The first run is configured with a population size of 25 and a maximum number of generations of 10 (total sample budget 250) and the second run is configured with a population size of 50 and a maximum number of generations of 50 (total sample budget 2500). The remaining parameters have been left to their default values.

C. Results

Results for the benchmark functions have been summarized in Table II. The EMO runs outperform NSGA-II and SPEA2 for each DTLZ function in terms of hypervolume score as well as number of samples. Surprisingly, the EMO ($\xi = 10^{-5}$) run achieves a higher hypervolume score than the EMO ($\xi = 0$) run. This can be explained by the fact that EMO ($\xi = 10^{-5}$) spends less samples on the small (but certain) improvements and more samples on the larger (less certain) improvements. For the benchmark functions considered in this paper the Kriging models are accurate enough to correctly predict the larger improvements. Evidently, this may not be the case for other optimization problems. A plot of the EI criterion is shown in Figure 4 for the DTLZ7 benchmark function.

Table II: Results of the EMO algorithm, NSGA-II and SPEA2. The EMO algorithm consistently outperforms the two MOEAs on the hypervolume for all benchmark functions. Furthermore, EMO ($\xi = 10^{-5}$) is slightly better than EMO ($\xi = 0$) which can be attributed to the fact that EMO ($\xi = 0$) makes small iterative improvements while for EMO ($\xi = 10^{-5}$) these improvements are not considered (see section III-C) and the focus is more on larger (uncertain) improvements.

	Algorithm	Sample budget	Initial hypervolume ($\xi = 0$)	Final hypervolume ($\xi = 0$)
DTLZ2	EMO ($\xi = 0$)	250	13.9649	14.9225
	EMO ($\xi = 10^{-5}$)		13.9649	**14.9439**
	NSGA-II		12.3514	13.6284
	SPEA2		unknown	14.4873
	NSGA-II	2500	13.5980	14.6953
	SPEA2		unknown	14.8503
DTLZ7	EMO ($\xi = 0$)	250	27.4022	42.7030
	EMO ($\xi = 10^{-5}$)		27.4022	**42.9865**
	NSGA-II		16.7547	18.0841
	SPEA2		unknown	37.4830
	NSGA-II	2500	23.4465	28.1785
	SPEA2		unknown	42.1191
DTLZ5	EMO ($\xi = 10^{-5}$)	250	193.5089	**196.8646**
	NSGA-II		178.5896	192.0376
	SPEA2		unknown	192.6617
	NSGA-II	2500	184.9897	193.9606
	SPEA2		unknown	194.3750

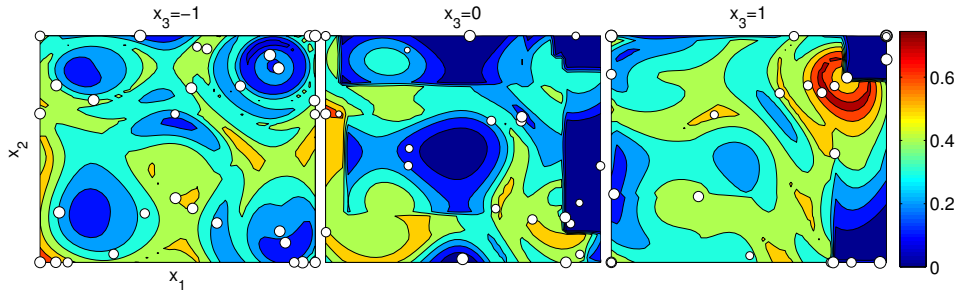


Figure 4: Contour plots of the expected improvement criterion for the DTLZ7 function (based on four Kriging models of 250 samples) where $x_3 = -1$, $x_3 = 0$ and $x_3 = 1$, respectively. The remaining variables are set fixed to $x_4 = x_5 = x_6 = -1$, i.e., the location of the real Pareto front. The dots represent (a projection of) the samples, the larger a dot the closer the sample is to the actual slice.

While the EMO algorithm clearly outperforms NSGA-II and SPEA2 in the number of expensive samples and hypervolume score there are some limitations. The EMO algorithm, and other MOSBO techniques, rely on the quality of the surrogate model to guide the selection of new expensive data points. While the Kriging models do not have to be accurate at the start of the algorithm when using the EI and PoI criterions, the Kriging models should be able to capture the objective functions accurately when enough samples become available, which might not always be the case. Furthermore, the calculation of the statistical criterions comes at a computational cost, similar to the computational cost of MOEAs that rely on the hypervolume, which might limit the practical usage of the EMO algorithm for some (less expensive) optimization problems.

A plot of the computation time of the branch-and-bound procedure versus the number of samples for the DTLZ7 function is shown in Figure 5. Initially, the computation time increases quite rapidly as the expected improvement criterion focuses on exploration which leads to the extension of the current, sub-optimal Pareto set. After enough samples have been obtained (at approximately 90 samples) the Kriging models are accurate enough to allow detection of points that improve on the Pareto set (exploitation). In effect, reducing the size of the Pareto set and, hence, decreasing the computation time of the branch-and-bound procedure. Only when the Pareto set is sufficiently improved, more points are selected that augment the Pareto set and the computation time increases again. Note that the computation time is often zero, meaning that the Pareto set has not been changed since the previous iteration and thus the integral bounds do not need to be recalculated.

V. Conclusion

The authors presented the Efficient Multiobjective Optimization (EMO) algorithm, which uses multiobjective versions of the Expected Improvement (EI) and Probability of Improvement (PoI) to identify the Pareto front with a limited sample budget. The EMO algorithm is compared against the well-known SPEA2 and NSGA-II evolutionary methods with promising results. In theory an arbitrary number of objective functions can be handled. However, in practice due to the nature of the multiobjective EI and PoI statistical criterions EMO also does not escape the curse of dimensionality (no-free-lunch theorem) with respect to the number of objective functions and number of Pareto points. The suggested method already circumvents the largest computation cost and can eas-

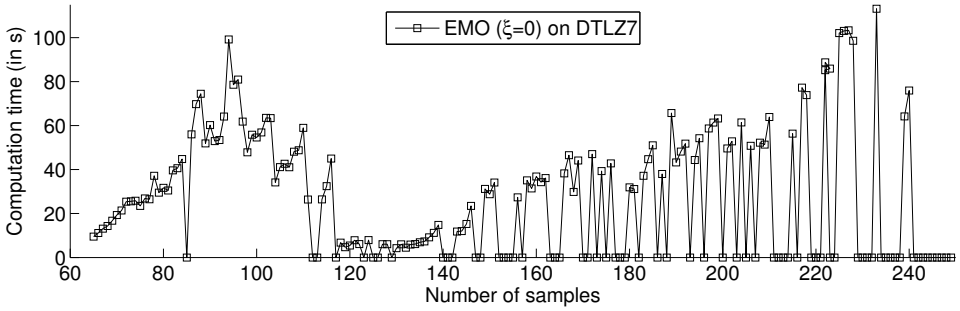


Figure 5: Computation time of the integral bounds versus the number of samples of the exact EMO ($\xi = 0$) run for the DTLZ7 function.

ily generate fast approximated versions of the multiobjective criterions with little loss of efficiency. This is illustrated in the benchmark examples.

Future work will focus on leveraging the ideas of several hypervolume calculation routines that exist in literature for a more efficient calculation of the EI and PoI criterions. Furthermore it may be useful to consider an update scheme for the integral bounds, which will be considerable more efficient than recalculating the integrals bounds almost each iteration. Indirectly, a speedup can also be achieved by selecting multiple update points at a time. In addition, the EMO algorithm will be compared against other MOSBO techniques and more recent MOEAs, e.g., SMS-EMOA. Finally, the effect of using the approximated statistical criterions will be further investigated.

ACKNOWLEDGMENTS

Ivo Couckuyt is funded by the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen). This work was supported by the Fund for Scientific Research in Flanders (FWO-Vlaanderen). Dirk Deschrijver is a post-doctoral research fellow of FWO-Vlaanderen.

REFERENCES

[1] G. Wang and S. Shan, "Review of metamodeling techniques in support of engineering design optimization," *Journal of Mechanical Design*, vol. 129, no. 4, pp. 370–380, 2007.

[2] D. Gorissen, K. Crombecq, I. Couckuyt, P. Demeester, and T. Dhaene, "A surrogate modeling and adaptive sampling toolbox for computer based design," *Journal of Machine Learning Research*, vol. 11, pp. 2051–2055, 2010. [Online]. Available: http://sumo.intec.ugent.be/

[3] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black-box functions," *J. of Global Optimization*, vol. 13, no. 4, pp. 455–492, 1998.

[4] J. P. C. Kleijnen, *DASE : Design and Analysis of Simulation Experiments*. Springer, May 2007.

[5] J. de Brabanter, "LS-SVM regression modelling and its applications," Ph.D. dissertation, Katholieke Universitiet Leuven, 2004.

[6] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *Evolutionary Computation, IEEE Transactions on*, vol. 6, no. 2, pp. 182–197, 2002.

[7] E. Zitzler, M. Laumanns, and L. Thiele, "Spea2: Improving the performance of the strength pareto evolutionary algorithm," Swiss Federal Institute of Technology, Tech. Rep., 2001.

[8] N. Beume, B. Naujoks, and M. Emmerich, "Sms-emoa: Multiobjective selection based on dominated hypervolume," *European Journal of Operational Research*, vol. 181, no. 3, pp. 1653–1669, 2007.

[9] Q. Zhang, W. Liu, E. Tsang, and B. Virginas, "Expensive multiobjective optimization by moea/d with gaussian process model," *Evolutionary Computation*, vol. 14, no. 3, pp. 456–474, 2010.

[10] A. Gaspar-Cunha and A. Vieira, "A multi-objective evolutionary algorithm using neural networks to approximate fitness evaluations," *International Journal of Computers, Systems, and Signals*, vol. 6, no. 1, pp. 18–36, 2004.

[11] I. Voutchkov and A. Keane, "Multiobjective Optimization using Surrogates," in *Adaptive Computing in Design and Manufacture 2006. Proceedings of the Seventh International Conference*, I. Parmee, Ed., Bristol, UK, Apr. 2006, pp. 167–175.

[12] J. Knowles and H. Nakayama, "Meta-modeling in multiobjective optimization," in *Multiobjective Optimization: Interactive and Evolutionary Approaches*. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 245–284.

[13] A. Zhou, B.-Y. Qu, H. Li, S.-Z. Zhao, P. N. Suganthan, and Q. Zhang, "Multiobjective evolutionary algorithms: A survey of the state of the art," *Swarm and Evolutionary Computation*, vol. 1, pp. 32–49, 2011.

[14] 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.

[15] J. Knowles, D. Corne, and A. Reynolds, "Noisy multiobjective optimization on a budget of 250 evaluations," in *5th International Conference on Evolutionary Multi-Criterion Optimization*, 2009.

[16] A. J. Keane, "Statistical improvement criteria for use in multiobjective design optimization," *AIAA Journal*, vol. 44, no. 4, pp. 879–891, 2006.

[17] E. Zitzler, L. Thiele, M. Laumanns, C. Fonseca, and V. G. da Fonseca, "Performance assesment of multiobjective optimizers: an analysis and review," *Evolutionary Computation*, vol. 7, no. 2, pp. 117–132, 2003.

[18] K. Deb, L. Thiele, M. Laummans, and E. Zitzler, "Scalable test problems for evolutionary multi-objective optimization," Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, Tech. Rep. 112, 2001.

[19] T. Santner, B. Williams, and W. Notz, *The design and analysis of computer experiments*, ser. Springer series in statistics. New York: Springer-Verlag, 2003.

[20] A. Forrester, A. Sobester, and A. Keane, *Engineering Design Via Surrogate Modelling: A Practical Guide*. Chichester: Wiley, 2008.

[21] D. R. Jones, "A taxonomy of global optimization methods based on response surfaces," *Global Optimization*, vol. 21, pp. 345–383, 2001.

[22] I. Couckuyt, F. Declercq, T. Dhaene, and H. Rogier, "Surrogate-based infill optimization applied to electromagnetic problems," *Advances in design optimization of microwave/rf circuits and systems (special issue)*, vol. 20, no. 5, p. 492, 2010.

[23] G. I. Hawke and J. K. Sykulski, "An enhanced probability of improvement utility function for locating pareto optimal solutions," in *16th Conference on the Computation of Electromagnetic Fields*, 2008.

[24] N. Beume, C. M. Fonseca, and M. L.-I. nez, "On the complexity of computing the hypervolume indicator," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 1075–1082, 2009.

[25] K. Bringmann and T. Friedrich, "An efficient algorithm for computing hypervolume contributions," *Evolutionary Computation*, vol. 18, no. 3, pp. 383–402, 2010.

[26] E. Dam, B. van Husslage, D. den Hertog, and J. Melissen, "Maximin Latin hypercube designs in two dimensions," *Operations Research*, vol. 55, no. 1, pp. 158–169, 2007.

[27] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[28] J. Bonnans, J. Gilbert, C. Lemaréchal, and C. Sagastizábal, *Numerical Optimization: Theoretical and Practical Aspects*. Springer, 2006.