



Optimistic NAUTILUS navigator for multiobjective optimization with costly function evaluations

Bhupinder Singh Saini¹ · Michael Emmerich^{1,2} · Atanu Mazumdar¹ · Bekir Afsar¹ · Babooshka Shavazipour¹ · Kaisa Miettinen¹

Received: 20 November 2020 / Accepted: 2 December 2021 / Published online: 3 January 2022
© The Author(s) 2021

Abstract

We introduce novel concepts to solve multiobjective optimization problems involving (computationally) expensive function evaluations and propose a new interactive method called O-NAUTILUS. It combines ideas of trade-off free search and navigation (where a decision maker sees changes in objective function values in real time) and extends the NAUTILUS Navigator method to surrogate-assisted optimization. Importantly, it utilizes uncertainty quantification from surrogate models like Kriging or properties like Lipschitz continuity to approximate a so-called optimistic Pareto optimal set. This enables the decision maker to search in unexplored parts of the Pareto optimal set and requires a small amount of expensive function evaluations. We share the implementation of O-NAUTILUS as open source code. Thanks to its graphical user interface, a decision maker can see in real time how the preferences provided affect the direction of the search. We demonstrate the potential and benefits of O-NAUTILUS with a problem related to the design of vehicles.

Keywords Interactive methods · Multiobjective optimization problems · Decision makers · Preference information · Computational cost · Kriging

1 Introduction

Multiobjective optimization deals with the simultaneous minimization or maximization of multiple conflicting objective functions. There, instead of a single optimal solution, so-called Pareto optimal solutions can be identified with different trade-offs. They form a Pareto optimal set. Typically, preference information from a domain expert, a decision maker (DM), is needed to identify the most preferred one among the mathematically incomparable Pareto optimal solutions.

✉ Bhupinder Singh Saini
bhupinder.s.saini@jyu.fi

¹ Faculty of Information Technology, University of Jyväskylä, P.O. Box 35 (Agora), FI-40014 Jyväskylä, Finland

² Faculty of Science, Leiden Institute of Advanced Computer Science, Niels Bohrweg 1, 2333 CA Leiden, The Netherlands

In multiobjective optimization, relevant questions include: How can we scale to problems with a large number of objectives? How can we solve problems with realistic, computationally expensive objective function formulations? How can an algorithm conveniently integrate a DM's preferences into the search? This article answers these questions by proposing a novel method called *Optimistic NAUTILUS Navigator method*, for short, O-NAUTILUS, which extends the interactive NAUTILUS Navigator method [38] to an *online data-driven approach* [18], sparingly needing new objective function evaluations during the interactive solution process and displaying additional information to support the DM. O-NAUTILUS combines two methodologies, which we briefly discuss next: surrogate modelling and navigation methods.

When dealing with computationally expensive problems, the computing resources are usually limited and one has to think about how to use them wisely. For instance, in engineering optimization, numerical simulation is often needed to compute the objective function values, and the evaluation of an objective function can take from several minutes up to hours [25]. One can reduce the computational cost by replacing the original (expensive) objective functions with computationally less costly approximation functions, typically learned from previous evaluation data. They are called *surrogate models* or *metamodels* [2, 19, 37, 39, 44, 46]. Different surrogate models have been developed in the literature (e.g., radial basis functions [36], neural networks [23, 32], support vector regression [3], polynomial regression [17] and Kriging [22, 25]) and utilized in various optimization methods. To get an overview of the available methods to handle computationally expensive multiobjective optimization problems, we refer to [43] (for exact and deterministic methods) and [7] (for evolutionary methods). To solve such problems, we cannot necessarily rely on mathematical properties such as differentiability or convexity. Metaheuristic approaches like evolutionary algorithms do not make any such assumptions [8]. However, neither their global nor local convergence can always be guaranteed [40].

An important aspect when using surrogate models is the handling of prediction uncertainty. Although replacing the original expensive functions with an approximated one is intended to reduce the computational cost of function evaluations, it often leads to a loss of accuracy. Approximations include some errors making the solutions inexact. Therefore, in multiobjective optimization with expensive black-box functions, we cannot always guarantee to reach actual Pareto optimality but can only compute approximations, particularly when we have a limited computation budget. Consequently, we cannot always explore all parts of the feasible region, and some Pareto optimal solutions which can be of interest to a DM may remain undiscovered. However, it is possible to estimate the range of improvements that may be achieved by further exploration using uncertainty quantification techniques. As a surrogate model, Kriging (Gaussian process regression) [22, 25, 33] is frequently used since it provides uncertainty quantification in the form of a local mean squared error in addition to the predicted value of the original function. Furthermore, some mathematical properties (such as Lipschitz continuity) may hold, which can be used to provide lower and upper bounds of function values at yet un-evaluated decision vectors. In the literature, Lipschitz continuity has been utilized in deterministic approaches which can guarantee the global convergence of solutions under certain conditions (e.g., [11, 16, 35, 39, 42, 46]). For Lipschitz continuous functions, lower and upper bounds can be relatively plainly calculated [24, 34, 48].

The second methodology in this article is navigation, a special type of interactive method [14]. As mentioned, preference information of a DM is typically needed to find the most preferred solution. We can classify multiobjective optimization methods based on when preference information is incorporated [27]. In *a priori* methods, the DM provides hopes and expectations first, and then a solution which matches them as well as possible is found, but

the hopes may be unrealistic. Alternatively, a representative set of Pareto optimal solutions is found, and then the DM must select the best of them in a *a posteriori* methods. However, generating a representative set may be computationally demanding and comparing many solutions cognitively demanding. Interactive methods aim to avoid these shortcomings.

In interactive methods [27,29,31], the DM takes part in the solution process iteratively and directs it with one's preference information. At the same time, (s)he learns about the problem, trade-offs involved and what kind of solutions are available. Thanks to learning, (s)he can also adjust preferences if so desired. Furthermore, only a limited amount of information needs to be processed at a time, which decreases cognitive load. The DM can provide preference information in different ways, for instance, as it is done in this paper, by providing aspiration levels representing desirable objective function values. These aspiration levels constitute a so-called reference point.

Many different interactive multiobjective optimization methods have been developed in the literature (see, e.g. [29,31] and references therein) and most of them deal with Pareto optimal solutions. Because of trade-offs between the objective functions, to achieve any improvement in one objective, the DM must sacrifice in some others, and this may hinder the DM's willingness to move. Accordingly, trade-off-free interactive methods such as the NAUTILUS family [30] have been proposed. They start from an inferior solution and iteratively approach Pareto optimal solutions by simultaneously improving all the objectives while following the DM's preferences. The solution process ends when a Pareto optimal solution is reached.

NAUTILUS Navigator [38] combines NAUTILUS ideas with navigation [14]. Supported by a visual user interface, the DM can navigate to see how objective function values evolve in real time and improve all objective values simultaneously. The DM directs the navigation with reference points and the search progresses towards them.

The contribution of this article is as follows. We focus at solving problems that contain computationally expensive objective functions. Our new method, O-NAUTILUS, uses surrogate models with uncertainty quantification, and can be applied in combination with both heuristic and deterministic optimization algorithms. In O-NAUTILUS, we consider the probability or possibility of extending the estimated Pareto optimal set, constructed by using surrogate models. In our proposed method, we use uncertainty quantification in the form of (confidence) bounds from Kriging or Lipschitzian models, to build and update an optimistic approximation of the Pareto optimal set and then use this information in the interactive method.

For biobjective problems, there have been some attempts in the literature to visually aid the DM in choosing a preferred solution using surrogate models in [47]. However, our proposed method is not limited to biobjective problems. Another significant difference is that we use the optimistic approximation as a part of an interactive method to aid the DM in making targeted function evaluations. Accordingly, the DM has an option to extend the search area and cross the current borders of the estimated Pareto optimal set for further discovery towards optimistic boundaries. This will trigger an exploration phase: new evaluations with the costly objective functions are conducted in a targeted way in order to assess possibilities to extend the Pareto optimal set and find an improvement in the preferred direction.

An important characteristic of O-NAUTILUS is the alternating phases in the algorithm which require the presence of a DM (computationally fast phase) and which do not require the presence of a DM (computationally expensive phase). The DM may use one's expertise to judge how long a single exact function evaluation would take. The DM's attention is not needed during the computationally expensive phase.

The rest of the paper is structured as follows. We provide some background material together with concepts and notations in Sect. 2. In Sect. 3, we introduce our new method,

O-NAUTILUS. We demonstrate the applicability of O-NAUTILUS, and compare it with NAUTILUS Navigator with a case study in Sect. 4. Finally, we conclude and mention future research directions in Sect. 5.

2 Background: basic concepts and notation

This section covers basic concepts and notation of multiobjective optimization, NAUTILUS methods and surrogate models needed in the rest of the paper. As said, we apply Kriging and Lipschitzian models as surrogate models.

2.1 Multiobjective optimization

We consider multiobjective optimization problems with $k \geq 2$ objective functions $f_i : S \rightarrow \mathbb{R}$

$$\begin{aligned} & \text{minimize } \{f_1(\mathbf{x}), \dots, f_k(\mathbf{x})\} \\ & \text{subject to } \mathbf{x} \in S, \end{aligned} \quad (1)$$

where vectors of decision variables (for short, *decision vectors*) $\mathbf{x} = (x_1, \dots, x_n)^T$ belong to the feasible set $S \subset \mathbb{R}^n$ in the decision space. We define *objective vectors* as vectors in the *objective space* \mathbb{R}^k that consist of objective function values $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \dots, f_k(\mathbf{x}))^T$. Here we assume that objective functions are continuous and their evaluations are expensive.

Because objective functions are typically conflicting with each other, it is not possible to find a solution with all objectives reaching their individual optima and thus we consider so-called Pareto optimal solutions. In them, no objective function value can be improved without impairment in at least one of the others. We say that $\mathbf{z}^{(1)}$ *dominates* $\mathbf{z}^{(2)}$ (written as $\mathbf{z}^{(1)} \succ \mathbf{z}^{(2)}$) with $\mathbf{z}^{(1)}, \mathbf{z}^{(2)} \in \mathbb{R}^k$, if $z_i^{(1)} \leq z_i^{(2)}$ for $i = 1, \dots, k$ and $z_j^{(1)} < z_j^{(2)}$ for at least one index j . If $\mathbf{z}^{(1)}$ and $\mathbf{z}^{(2)}$ do not dominate each other, they are called mutually nondominated. Furthermore, a decision vector $\mathbf{x}^* \in S$ and the corresponding objective vector $\mathbf{f}(\mathbf{x}^*)$ are called Pareto optimal, if there does not exist another $\mathbf{x} \in S$ such that $\mathbf{f}(\mathbf{x})$ dominates $\mathbf{f}(\mathbf{x}^*)$. Typically, problem (1) has many Pareto optimal solutions constituting a so-called Pareto optimal set E . Its image in the objective space is called a Pareto (optimal) front $\mathbf{f}(E)$. Here, we refer to objective vectors that are mappings of decision vectors as solutions. In addition, we call vectors in the objective space without any corresponding decision vector as points.

Usually, we need a DM with domain expertise to decide which Pareto optimal solution is the most preferred one satisfying her/his preferences. We denote it as \mathbf{z}_{pref} . An analyst can also take part in the solution process. By an analyst we refer to a human or a computer program supporting the DM and typically taking care of mathematical aspects.

Information about the ranges of objective function values in the Pareto front can be useful for the DM. The best individual optima are components of an *ideal point* $\mathbf{z}^* = (z_1^*, \dots, z_k^*)^T$ with $z_i^* = \min_{\mathbf{x} \in S} f_i(\mathbf{x}) = \min_{\mathbf{x} \in E} f_i(\mathbf{x})$ for $i = 1, \dots, k$. The worst values represented in a *nadir point* $\mathbf{z}^{\text{nad}} = (z_1^{\text{nad}}, \dots, z_k^{\text{nad}})^T$ with $z_i^{\text{nad}} = \max_{\mathbf{x} \in E} f_i(\mathbf{x})$ for $i = 1, \dots, k$ are in practice difficult to calculate because the set E is unknown. The nadir point can be approximated (see, e.g., [27] and references therein). It is also possible to ask the DM to provide the worst possible objective function values (s)he can think of and constitute a nadir point of them.

An important concept in this paper is reachability. For a point $\mathbf{z} \in \mathbb{R}^k$, if $\mathbf{f}(\mathbf{x})$ dominates \mathbf{z} , we say that $\mathbf{x} \in S$ is reachable from \mathbf{z} . Furthermore, we define a reachable region as a subset

of decision vectors in the Pareto optimal set which are reachable from \mathbf{z} . In addition, the image of the reachable region from \mathbf{z} in \mathbb{R}^k is also called a reachable region and it contains all objective function values which can be reached from \mathbf{z} . In the following, we assume that \mathbf{z} is clear from the context. Therefore, we shorten the term as a reachable region without explicitly mentioning \mathbf{z} .

As mentioned in the introduction, an example of providing preference information is a reference point $\mathbf{q} = (q_1, \dots, q_k)^T$ consisting of desirable values of each objective function provided by the DM. If the individual values q_i can be simultaneously achieved or improved in a feasible solution, the reference point is called achievable and if this is not the case, it is called unachievable.

Scalarization functions such as an achievement scalarization function (ASF) [27] can be used for solving multiobjective optimization problems in an interactive way. For a reference point \mathbf{q} , the ASF can be defined as:

$$s^z(\mathbf{f}(\mathbf{x})) = \max_{i=1, \dots, k} \left[\frac{f_i(\mathbf{x}) - q_i}{z_i^{\text{nad}} - z_i^{**}} \right] + \rho \sum_{i=1}^k (f_i(\mathbf{x}) - q_i), \tag{2}$$

where ρ is a small, positive augmentation coefficient, $z_i^{**} = z_i^* - \epsilon$ ($i = 1, \dots, k$) are components of a so-called utopian point $\mathbf{z}^{**} \in \mathbb{R}^k$ and $\epsilon > 0$ is a small scalar. With preferences given as a reference point \mathbf{q} , we can get a Pareto optimal solution to problem (1) by minimizing the ASF in (2); see, e.g. [27,45] for details.

2.2 Overview of NAUTILUS family

As mentioned in the introduction, the idea of the interactive methods in the NAUTILUS family [30] is to enable the DM in identifying one’s most preferred solution by starting from a bad solution (like a nadir point) and proceeding iteratively by gaining improvement in all objectives simultaneously. In this way, the DM receives solutions that dominate each other from one iteration to another and gets a Pareto optimal solution only at the end of the solution process. By avoiding the need of trading off between the objectives, the DM can reach any Pareto optimal solution [28].

When applying interactive methods that operate with Pareto optimal solutions throughout the solution process, the DM has to allow sacrifices in at least one objective function to find a new Pareto optimal solution. This may hinder the DM’s willingness to move, referred to as anchoring [4]. Furthermore, according to the prospect theory [21], past experiences affect people’s hopes and expectations, and we do not react symmetrically to gains and losses. Because of this, the DM may converge prematurely and fail to find one’s most preferred solution.

All methods in the NAUTILUS family enable the DM to freely focus on the part of the Pareto front that is interesting without making sacrifices. Family members differ from each other in the way the DM provides preference information to direct the solution process and how solutions are generated from iteration to iteration. These differences are described as a NAUTILUS framework in [30].

Since the DM gradually approaches the Pareto front, the reachable region, that is, the part of the Pareto front that still can be reached without trading off, shrinks. In other words, there are other parts of the Pareto front that can only be reached if the DM goes backwards and, in that way, widens up the reachable region.

The latest member of the NAUTILUS family is NAUTILUS Navigator [38]. With it, the DM can navigate in the reachable region in real time and improve simultaneously all objective values. The information shown to the DM is a visual presentation of how the reachable ranges shrink when one approaches the Pareto front.

To be more specific, the reachable ranges describe intervals of objective function values in the subset of Pareto optimal solutions which still are reachable from the current point without trading off. If the solution process starts from the nadir point, the reachable range of each objective function is defined at the beginning with the ideal and nadir points. When the solution process continues, the ranges are defined by the so-called current iteration point and the point with the best reachable values.

To get started, NAUTILUS Navigator needs a set of solutions that approximate the Pareto optimal front. They are all assumed to be mutually nondominated. Besides, during the navigation, we lose the connection to the decision space. This is not a problem because we can guarantee that at the end of the navigation process, the DM will reach a nondominated solution and can find the corresponding decision vector in the decision space. For details of the method, see [38].

2.3 Surrogate models

In the following, by exact objective function evaluations, we mean the evaluation of the objective functions in (1). For the O-NAUTILUS method, we utilize methods that can predict function values at yet un-evaluated decision vectors $\mathbf{x}^+ \in S$ utilizing a set of N already evaluated decision vectors, say $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(N)}$, with $\mathbf{y}^{(1)} = \mathbf{f}(\mathbf{x}^{(1)}), \dots, \mathbf{y}^{(N)} = \mathbf{f}(\mathbf{x}^{(N)})$. In multiobjective optimization, a common strategy is to train a surrogate model for each of the objective functions f_i separately. We will denote the predictions of function values with $\hat{y}_i \approx f_i(\mathbf{x}^+)$ and the corresponding exact function values with $y_i = f_i(\mathbf{x}^+)$.

Furthermore, we need methods that can assess the uncertainty of the predictions by providing an uncertainty quantification in the form of ranges in which the true outcome is (likely) to be found. Kriging models and Lipschitzian models are two common classes of such surrogate models, which is the reason why we chose them as surrogate models in our discussion. The upper bound of these ranges will be denoted with $\bar{f}_i(\mathbf{x}^+)$, and the lower bound with $\underline{f}_i(\mathbf{x}^+)$. The ranges may have a probabilistic interpretation, such as in the Kriging models, or a possibilistic, exact interpretation, such as in the Lipschitzian models.

2.3.1 Kriging and Gaussian process regression

The Kriging method and Gaussian process regression are mathematically very similar¹. To obtain a prediction, function values at neighboring decision vectors of the new decision vector are weighted by distance and a factor that is determined in a training process. While the training can be time-consuming, predictions and uncertainty quantification for a new decision vector are very fast for such vectors that have not been evaluated yet. Therefore, in order to find promising regions for new evaluations, a Kriging model can be evaluated in many different decision vectors. As mentioned in the general introduction to surrogate models, also in the Kriging methods we handle multiple objectives typically by learning them separately

¹ Kriging (named after geo-scientist Krige) seeks to find a best linear unbiased predictor assuming the observed data is a realization of a stochastic processes or random field (not necessarily of the Gaussian type). Gaussian process regression is motivated by Bayesian reasoning and uses conditional mean and variance of Gaussian random fields to model and bound the objective function at a given decision vector.

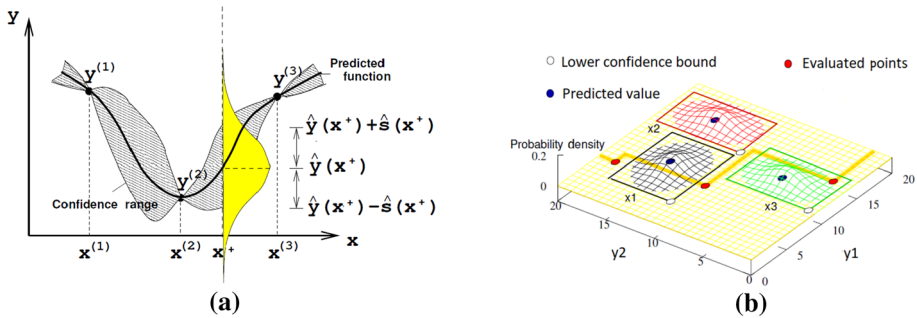


Fig. 1 Kriging prediction in **a** 1-D and **b** 2-D objective spaces

for each objective function [9]. The predictive distribution at a new, yet un-evaluated decision vector \mathbf{x}^+ is then given by an independent multivariate normal distribution for each decision vector with mean values $\hat{y}_i(\mathbf{x}^+)$ and standard deviations $\hat{s}_i(\mathbf{x}^+)$ for the objective functions $f_i, i = 1, \dots, k$. Based on this, we can compute probabilistic confidence ranges with a lower bound $\underline{f}(\mathbf{x})$ and an upper bound $\overline{f}(\mathbf{x})$, respectively, defined for $i = 1, \dots, k$ as:

$$\underline{f}_i(\mathbf{x}^+) = \hat{y}_i(\mathbf{x}^+) - \alpha \hat{s}_i(\mathbf{x}^+) \tag{3}$$

$$\overline{f}_i(\mathbf{x}^+) = \hat{y}_i(\mathbf{x}^+) + \alpha \hat{s}_i(\mathbf{x}^+), \tag{4}$$

where $\alpha \geq 0$ is a user-defined confidence level.

An illustration of a Kriging model for a 1-D decision vector and a single objective function is provided in Fig. 1a and for a biobjective problem in Fig. 1b. In Fig. 1a, the vertical axis (y) denotes the function values obtained at three decision vectors, namely $y^{(1)} = f(\mathbf{x}^{(1)})$, $y^{(2)} = f(\mathbf{x}^{(2)})$ and $y^{(3)} = f(\mathbf{x}^{(3)})$. The predictions at a new decision vector \mathbf{x}^+ are given by a 1-D normal distribution with a mean value $\hat{y}(\mathbf{x}^+)$ and a standard deviation $\hat{s}(\mathbf{x}^+)$ quantifying the uncertainty of the prediction. Figure 1b illustrates three predictions for a biobjective problem. Again, the uncertain outcome of the expensive evaluations is quantified by means of normal distributions that are indicated in the figure by their probability density function (PDF). In the figure, we deal with bi-variate distributions. They can be used to estimate 2-D confidence ranges which are indicated in the figure by rectangles. From these, it is straightforward to compute optimistic bounds for the objective vector resulting from an evaluation at a specific decision vector.

The Kriging or Gaussian process method underlies several statistical assumptions, most importantly a distance based correlation between outputs at decision vectors, where the distance is measured in the decision space. For an in-depth description of the Kriging method and its statistical motivation the reader is referred to [41]. In our experiments we use a standard implementation of Kriging with an isotropic, exponential kernel.

2.3.2 Lipschitz bounds for prediction and uncertainty quantification

The knowledge of the Lipschitz constant can help in designing global search algorithms [24]. Lower or upper bounds (also called shells) for the values of a Lipschitz continuous function can then be relatively simply computed and have been used to construct such global optimization algorithms [24,34,48]. As will be shown, using a Lipschitz constant will also yield an alternative, yet linear approach to surrogate modeling with uncertainty quantification in the form of a confidence range. As it is also based on distances in the decision space and

Fig. 2 Computation of Lipschitz bounds in one dimension

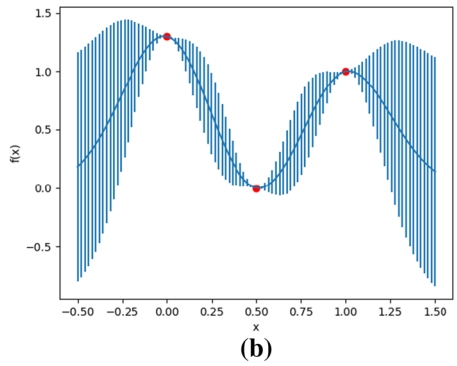
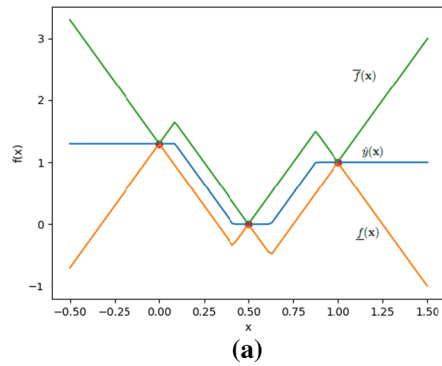
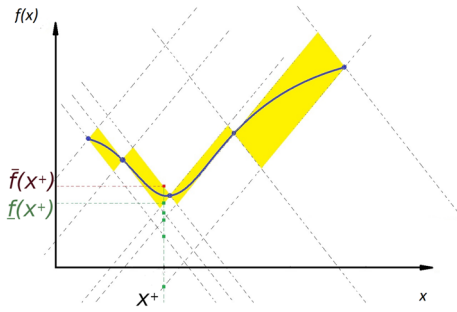


Fig. 3 Lipschitz envelope for a 1-D function (a). Kriging uncertainty ranges for the same data (b)

provides confidence bounds, it is structurally quite similar to the aforementioned Kriging method.

A function f is called Lipschitz continuous if there exists a real positive constant L (called Lipschitz constant) such that for all $\mathbf{x}, \mathbf{x}' \in S$

$$d(f(\mathbf{x}), f(\mathbf{x}')) \leq Ld(\mathbf{x}, \mathbf{x}'),$$

where $d : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$ is a distance function. Let us again assume we are given a data set (set of evaluations) with N evaluated points (decision vectors). Then, if \mathbf{x}^+ denotes an un-evaluated decision vector, its output $y^+ = f(\mathbf{x}^+)$ is bounded by the interval $[\underline{f}(\mathbf{x}^+), \overline{f}(\mathbf{x}^+)] \subseteq \mathbb{R}$, with

$$\underline{f}_i(\mathbf{x}^+) = \max_{t=1, \dots, N} \{y_i^t - L_i d(\mathbf{x}^+, \mathbf{x}^{(t)})\} \text{ and} \tag{5}$$

$$\overline{f}_i(\mathbf{x}^+) = \min_{t=1, \dots, N} \{y_i^t + L_i d(\mathbf{x}^+, \mathbf{x}^{(t)})\}. \tag{6}$$

An example of the computation of Lipschitz bounds using these equations is given in Fig. 2. Moreover, we can define a prediction as the average of the upper and lower bound, i.e. $\hat{y}(\mathbf{x}^+) = \frac{1}{2}(\overline{f}(\mathbf{x}^+) + \underline{f}(\mathbf{x}^+))$, thus creating a Lipschitzian model. An illustration of the Lipschitzian envelope is given in Fig. 3a and a figure of the same envelope using Kriging is provided in Fig. 3b. The example data is part of an interactive Python plot, that can be accessed and modified at <https://trinket.io/python3/c38e5ebdbc>. Since the distance function can also be defined for multivariate decision spaces, the Lipschitzian bound calculation can also be used efficiently for high-dimensional decision spaces.

3 O-NAUTILUS method

In this section, we introduce the details of the O-NAUTILUS method. To be able to describe the O-NAUTILUS algorithm, we first need to introduce the major components and concepts utilized in the method. Hence, we describe them first and then provide a detailed pseudo-code of the algorithm.

The starting point of O-NAUTILUS is a pre-generated set of solutions. We do not make any specific assumptions regarding this set. It can, for example, be a rough approximation of the Pareto front, or even a space filling sample of solutions. The O-NAUTILUS method extends the NAUTILUS Navigator method by visualizing not just the reachable ranges, but also optimistic ranges, which represent solutions which are predicted to have good objective values, but are not represented in the set of known solutions. The O-NAUTILUS method does this by working with two sets of fronts at once. The first set is the nondominated front from the set of known solutions, the same as the one used in NAUTILUS Navigator. The second set is an optimistic estimation of the Pareto front given by a set of points in the objective space that is calculated by multiobjective minimization on the lower bounds $\underline{f}_i(\mathbf{x})$, $\mathbf{x} \in S$, $i = 1, \dots, k$ that are estimated using the surrogate model. We call it an optimistic front to be described further in the next subsection.

Using the information provided by the optimistic front and the corresponding ranges enables the DM to strategically conduct function evaluations. In terms of navigation, this means going beyond the reachable ranges and stepping into an “optimistic area” by conducting function evaluations that are likely to find solutions in that area. This has a two-fold benefit. Firstly, the function evaluations are conducted in regions of interest of the DM. Hence, no resources are wasted in finding solutions that may not be of interest to the DM. Secondly, the newly evaluated decision vectors can be added back to the known set of solutions. Based on this new known set, surrogate models can be trained again. As this new known set contains solutions in the region of interest of the DM, the surrogate models themselves perform better in the region of interest. This means that the optimistic predictions obtained by the models are more accurate in the region of interest. Hence, as the algorithm continues, the DM gets an increasingly improved picture of the objective space in the region of interest.

In Sects. 3.1 through 3.4, we describe various components of the O-NAUTILUS method. These components are modular and can be trivially replaced by alternatives which serve a similar purpose. Section 3.5 then describes the O-NAUTILUS method, which puts together the aforementioned components to support a DM to identify the most preferred solution.

3.1 Optimistic pareto Front from surrogate models

As described in (3) and (5), Kriging and Lipschitzian models, respectively, can be used to estimate or determine an optimistic lower bound for an objective f_i , $i = 1, \dots, k$, at any point in the decision space $\mathbf{x} \in S$ as $\underline{f}_i(\mathbf{x})$. To get an optimistic Pareto front from the surrogate models, we simply solve the following problem:

$$\begin{aligned} & \text{minimize } \{\underline{f}_1(\mathbf{x}), \dots, \underline{f}_k(\mathbf{x})\} \\ & \text{subject to } \mathbf{x} \in S, \end{aligned} \quad (7)$$

with any appropriate solver.

Figure 4 shows the concept of an optimistic front (for a biobjective example) graphically. The blue (darker in greyscale) points belong to a set of exactly evaluated decision vectors P . The orange (lighter in greyscale) points belong to the set of solutions obtained by solving

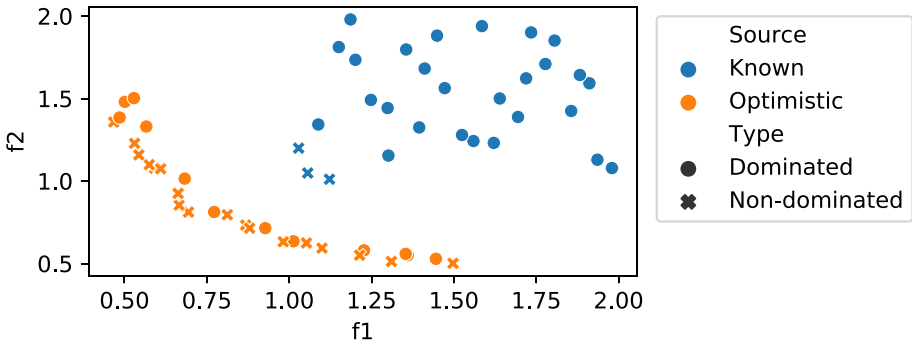


Fig. 4 Simplified figure to demonstrate the optimistic front

(7), to be denoted by P^+ . The nondominated points from P form the *known front*, whereas the nondominated points from P^+ form the *optimistic front*. These are represented as blue (darker) crosses (known front) and orange (lighter) crosses (optimistic front) in Fig. 4. These fronts will be used by the other steps of the O-NAUTILUS method.

3.2 Reachable ranges

In Fig. 5, a path of the O-NAUTILUS method is visualized in the objective space as a 2-D scatter plot (5a). The blue (darker) points in Fig. 5a represent P , whereas the orange (lighter) points represent P^+ . The path is also visualized as a reachable ranges plot (5b) which shows the reachable ranges for various objectives at different steps [38]. The vector $\mathbf{z}^{(i)}$ in the objective space is the *step point* at the i^{th} step and has a function similar to the *current point* in [38].

The intention of this visualization is to show how for every step point the reachable ranges change and how this can be visualized in the 2-D objective space plot (5a) and in the reachable ranges plot (5b). Note that the objective space visualization becomes impractical in higher dimensions, whereas the reachable ranges plot can still be used.

For the j^{th} objective, the *known reachable range* at step i is defined as:

$$\left[\min_{y \in P, y > \mathbf{z}^{(i)}} y_j, \max_{y \in P, y > \mathbf{z}^{(i)}} y_j \right], \tag{8}$$

whereas the *optimistic reachable range*, which is newly introduced in this paper, is defined by replacing P by P^+ in (8). These values are displayed in the *reachable range paths* as described in [38].

Focusing on the visualization of the path, we here omit details of the algorithmic procedure and interaction with the DM, which will be discussed in Sect. 3.3. The path starts in Fig. 5 at point ①. In the first step, the DM aims at merely improving f_2 which leads to ②. Then, the subsequent moves ②–⑤ are conducted such that f_1 and f_2 are equally improved. The last move ⑤–⑥ steps into the ‘orange (lighter) region’ where additional exploration in terms of exact function evaluation of the objective functions in the region of interest are required in order to assess the feasibility of the move. The fact that the move does not step beyond the orange region is indicating that such explorations will be promising and have a realistic chance of success.

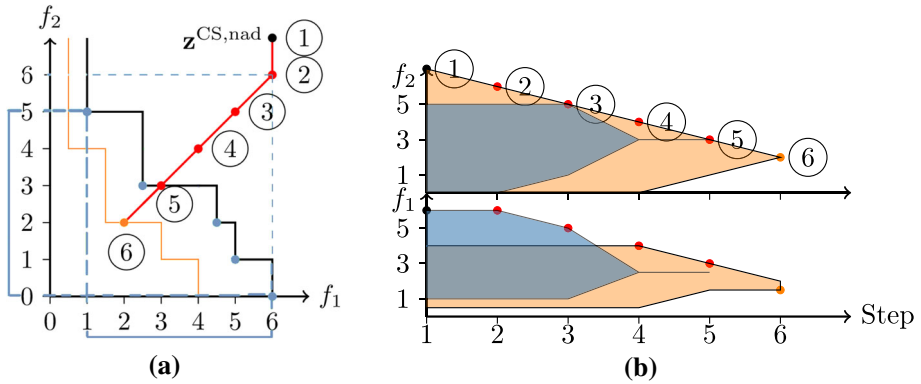


Fig. 5 Path of O-NAUTILUS visualized as a 2-D scatter plot (a) and as reachable ranges plot (b)

Looking at the relation between Fig. 5a (coordinate plot) and Fig. 5b (reachable ranges plot), let us focus on a single point on the path, say ②. As improvement is expected in all objectives, the maximal improvement of this point for which we still can guarantee the existence of a solution is indicated by the blue dashed lines (5a). This equates to values ranging from 1 to 6 units for f_1 and from 0 to 5 units for f_2 , as shown by the span along the two axes in Fig. 5a. The blue (darker) ranges in Fig. 5b show the same span at step ②. The orange (lighter) optimistic range, which extends the lower bound of the blue (darker) range, indicates how much we can still realistically expect to maximally improve by further exploration. This range is determined by the optimistic front.

3.3 Navigation

O-NAUTILUS uses the concept of navigation to help a DM make function evaluations at regions of interest and arrive at a desirable solution. Similar to NAUTILUS Navigator, the navigation begins at the worst possible objective values, a point that we will call a *combined set nadir point* ($z^{CS,nad}$). This point is calculated as the supremum of the combined set of known and optimistic fronts. The DM then takes a “step” by advancing the step point towards the solutions in any preferred direction, thus, gaining in each objective without any trade-offs. Unlike NAUTILUS Navigator, however, the DM does not reach a solution on the known front at the end of the navigation. This is because, unlike NAUTILUS Navigator, which uses an unchanging front, both the known and the optimistic fronts in O-NAUTILUS can change with further exact function evaluations.

Instead, the navigation is conducted in the following way. Firstly, a *combined set ideal point* ($z^{CS,*}$) is defined as the infimum of the combined set of known and optimistic fronts. From this, the *combined set utopian point* ($z^{CS,**}$) is generated in a corresponding manner to how the utopian point is derived from the ideal point. The *combined set nadir point* is calculated as described in the previous paragraph. Then, a hyperbox is formed with the combined set utopian and nadir points as the opposing corners. This hyperbox is divided using equidistant “rungs” perpendicular to the line connecting the combined set utopian and nadir points, starting and ending on those points, as shown in Fig. 6. The step point is then constrained to be inside the hyperbox and on one of these rungs throughout the navigation process. The number of these rungs is one more than the total number of steps to be taken during the navigation process, and it is pre-defined by an analyst. A higher step count divides

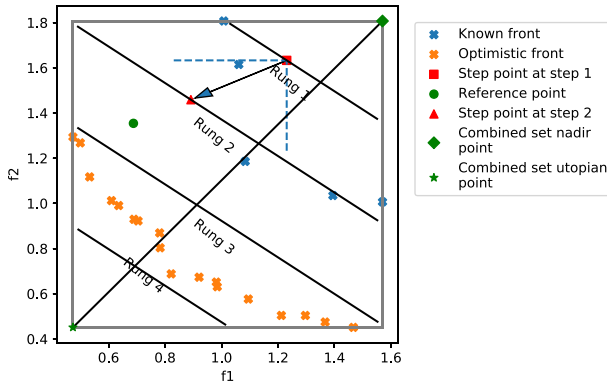


Fig. 6 Progressing the step point from step 1 to step 2 in the O-NAUTILUS algorithm. The step point (red square) jumps from rung 1 to rung 2 in the direction of the reference point (green circle). Here, combined set nadir and utopian points act as the zeroth and fifth rung, respectively

the hyperbox into smaller sections, giving the algorithm a higher resolution. This gives the DM a finer control over the navigation process.

The navigation begins at the step point at the zeroth rung, i.e. the combined set nadir point. The DM provides preference information that is used to define the direction of navigation (and hence, improvement). At any step, preference information (given in the form of aspiration levels) is valid if the corresponding reference point dominates the current step point. This is shown graphically in Fig. 6, for step 1. The step point is represented as the red square point on “Rung 1”. The reference point given, shown as a green circle, dominates the step point.

Once valid preference information is provided, the step point moves from the current rung to the next, in the direction of the reference point, as shown by the arrow in Fig. 6. The step points then keep jumping on the successive rungs in the same direction at a rate that can be controlled by a DM or an analyst. At every step, the reachable ranges are calculated and shown to the DM. The DM can update preference information, i.e., provide a new reference point at any step. Note that the basic unit of O-NAUTILUS is called a “step”, as opposed to “iteration”, which is the terminology used in most other interactive methods. In them, two iterations are typically separated by a DM providing new preference information. On the other hand, the steps in O-NAUTILUS proceed at a constant rate even if no new preference information is obtained from the DM after the first one. The rate can have a default value.

The following formula is used to calculate the successive step points:

$$\begin{aligned}
 \mathbf{z}^{(i+1)} &= \mathbf{z}^{(i)} + ss^{(i)} \mathbf{sd}^{(i)} \\
 \mathbf{sd}^{(i)} &= \frac{(\mathbf{z}^{\text{pref}} - \mathbf{z}^{(i)})}{\|\mathbf{z}^{\text{pref}} - \mathbf{z}^{(i)}\|} \\
 ss^{(i)} &= \left(\frac{\|\mathbf{z}^{\text{CS},**} - \mathbf{z}^{\text{CS},\text{nad}}\|}{i_{\text{total}}} \right)^2 \bigg/ \frac{(\mathbf{z}^{\text{pref}} - \mathbf{z}^{(i)}) \cdot (\mathbf{z}^{\text{CS},**} - \mathbf{z}^{\text{CS},\text{nad}})}{i_{\text{total}} \|\mathbf{z}^{\text{pref}} - \mathbf{z}^{(i)}\|} ,
 \end{aligned} \tag{9}$$

where i_{total} is the pre-defined maximum number of steps, \mathbf{z}^{pref} is a valid reference point, $\mathbf{sd}^{(i)}$ is the step direction and $ss^{(i)}$ is the step size at the i^{th} step (length of the arrow in Fig. 6). The \cdot symbol is used to denote the dot product of vectors, whereas $\|\cdot\|$ denotes the magnitude or ℓ_2 norm of a vector.

The navigation continues as long as there are known solutions reachable from the succeeding rung. At the end of these steps, the DM has a few options. (S)he can choose the last remaining known solution as the final solution, or restart the navigation process and navigate in a different direction to explore the two fronts. These options are available in NAUTILUS Navigator as well. In addition to these options, O-NAUTILUS provides the option to conduct a function evaluation at a single point. The DM, utilizing the information provided by the optimistic regions in the reachable ranges path, can decide whether to conduct an exact function evaluation to find a solution in the current region of interest (the optimistic regions of the reachable ranges plot). The mechanism to find such a potential solution to be evaluated is described in Sect. 3.4.

Based on the results of the function evaluation, the DM can choose to end the solution process, choosing the newly evaluated solution as the final solution. Alternatively, the DM can continue the search for alternative solutions. This is done by including the new solution in the known set of solutions P . Based on this updated set, new (and more accurate) surrogate models are trained. A new set of optimistic solutions P^+ is calculated as described in Sect. 3.1. The navigation is then restarted with the step point at the combined set nadir point, and with the last known aspiration levels as preference information. The DM can thus follow the same path travelled in the previous navigation phase to see how the reachable ranges have changed, or change preferences and follow a new path.

3.4 Expected ASF

Solving a surrogate assisted multiobjective optimization problem efficiently requires updating the surrogate models by utilizing an infill criterion. An infill criterion determines where the next sample is to be evaluated for updating the surrogate models. The infill criterion is obtained by optimizing an acquisition function that provides a mapping from a decision vector to a scalar value. In the literature, different acquisition functions have been suggested, i.e. expected improvement (EI) [20] and expected hypervolume improvement (EHVI) [10]. Both represent a trade-off between exploration and exploitation. The multiplicative EI (mEI) [12] is interesting in the context of the O-NAUTILUS method because it also takes into account a reference point provided by the DM. However, in our tests, solutions produced by mEI did not follow the DM’s preferences very well.

We propose an infill criterion called expected ASF (eASF) which is the expected value of (2). We use Monte Carlo sampling [15] to find the expected ASF. For a decision vector \mathbf{x} , we sample N_S points using the distribution predicted by the surrogate models. For example, while using Kriging surrogates, we use a normal distribution and the multivariate Gaussian PDF:

$$PDF_f^{Kriging} = \prod_{i=1}^k \frac{1}{\hat{\sigma}_i(\mathbf{x})\sqrt{2\pi}} \exp\left(-\frac{(\hat{y}_i(\mathbf{x}) - y_i)^2}{2\hat{\sigma}_i(\mathbf{x})^2}\right). \tag{10}$$

Here it can be observed that the distribution is Gaussian for Kriging surrogates. However, in case of Lipschitzian surrogates, we use a multivariate uniform distribution as the PDF to draw the samples. The set of samples $\{\hat{\mathbf{y}}_1(\mathbf{x}), \dots, \hat{\mathbf{y}}_{N_S}(\mathbf{x})\}$ that is drawn using (10) for a decision vector \mathbf{x}^+ is then used to calculate the ASF using (2). The set of ASF values is $\xi^z(\mathbf{x}) = \{s_1^z(\mathbf{x}), \dots, s_S^z(\mathbf{x})\}$. The final acquisition function is expected ASF or $E[\xi^z(\mathbf{x})]$. To find the infill point we solve the following single objective optimization problem:

$$g_{ASF}(\mathbf{x}) = \text{minimize } \{E[\xi^z(\mathbf{x})]\}$$

by using an appropriate optimization method. As the expected ASF considers the distribution of the ASF, it takes exploration in the search into account along with exploitation.

3.5 Algorithm description

Algorithm 1: O-NAUTILUS Algorithm

Input: Problem definition(1) MOP , set P of decision vectors and corresponding objective vectors, set of surrogate models SMT , multiobjective optimization algorithm (MOA) and function evaluation budget B .

```

1  $b \leftarrow 0$  // Function evaluation counter
2  $s_1, \dots, s_k \leftarrow \text{Train}(SMT, P)$  // Surrogate-models
3  $P^+ \leftarrow \text{Optimistic\_Optimize}(MOA, s_1, \dots, s_k)$ 
4 if  $\text{Function\_Evaluations\_Needed}(P, P^+)$  then
5 |  $P \leftarrow P \cup \text{Individual\_Optima}(s_1, \dots, s_k, MOP)$ 
6 |  $b \leftarrow b + k$  //  $k$  exact objective function evaluation used
7 | Go to step 2
8
9  $z^{CS,**}, z^{CS,nad} \leftarrow \text{Calculate\_Utopian\_And\_Nadir}(P, P^+)$ 
10  $i \leftarrow 0$  // Step number
11  $z^{(i)} \leftarrow z^{CS,nad}$  // Step point
12  $\text{Display\_Reachable\_Ranges}(P, P^+, z^i)$ 
13 if  $DM$  provides new preference or  $i = 0$  then
14 |  $z^{pref} \leftarrow \text{Get\_Preference\_From\_DM}()$ 
15
16 if  $DM$  wants to stop then
17 | Go to step 29
18
19 if Front  $P$  is not reached then
20 |  $z^{(i)} \leftarrow \text{Compute\_Next\_Iteration\_Point}(z^{(i)}, z^{pref})$ 
21 |  $i \leftarrow i + 1$ 
22 | Go to step 12
23 if  $b < B$  and  $DM$  wants to conduct exact function evaluation then
24 |  $P \leftarrow P + \text{Max\_Expected\_ASF}(z^{(i)}, s_1, \dots, s_k)$ 
25 |  $b \leftarrow b + 1$  // 1 exact point evaluation used
26 | Go to step 2
27
28 else
29 |  $\text{Display\_Chosen\_Solutions}(P, z^{(i)})$ 
30 end

```

Next, the complete interactive O-NAUTILUS algorithm is described. We pay particular attention to the use of exact objective function evaluations and the use of surrogate function evaluations, because the number of exact objective function evaluations will govern the total computational effort. The flow of the method is given in Algorithm 1. The various functions and variables involved in the algorithm are as follows:

1. Train takes as its input the choice of surrogate modeling technique (SMT) and the known set of solutions (P), and returns k trained surrogate models s_1, \dots, s_k , one for each of the k objectives. Here, SMT can be any surrogate modeling technique capable of giving optimistic predictions. In this paper, as mentioned, we consider the Kriging and Lipschitzian surrogate modeling techniques.

2. `Optimistic_Optimize` uses a multiobjective optimization method, in our implementation an evolutionary algorithm, with the surrogate models s_1, \dots, s_k to find an optimistic Pareto front (P^+) for the problem as described in Sect. 3.1. Note that no exact function evaluations are conducted in this step.
3. `Function_Evaluations_Needed` compares P and P^+ to determine whether further exact function evaluations are needed before the navigation and the involvement of the DM begins.
This need may arise, for example, if all solutions in P are dominated by the nondominated solutions in P^+ . This may happen because of two reasons. Firstly, the solutions in P may be far from the exact Pareto front of the problem. Alternatively, the surrogate models may not provide good predictions in certain regions, especially near the Pareto front. Either case may give misleading information to the DM when (s)he is asked to provide preferences. The first case can be resolved by conducting further exact function evaluations, preferably closer to the front. This will also lead to the generation of more samples for training the surrogate models, resolving the second case. There may be other cases where further exact function evaluations are desirable. The choice is left up to the analyst and not included in the algorithm.
4. `Individual_Optima` uses the surrogate models s_1, \dots, s_k to find k solutions corresponding to the maximum expected ASF (eASF) of the individual surrogate models. These solutions are then evaluated using the exact objective functions. Other strategies may be used in place of `Individual_Optima` to find good solutions to evaluate. For example, an alternative option is to use a few representative solutions from P^+ .
5. `Calculate_Utopian_And_Nadir` combines the nondominated solutions from the sets P and P^+ and returns the *combined set utopian point* ($\mathbf{z}^{\text{CS,**}}$) and the *combined set nadir point* ($\mathbf{z}^{\text{CS,nad}}$) of this combined set.
6. `Display_Reachable_Ranges` uses P , P^+ and $\mathbf{z}^{(i)}$ to calculate and display the known and optimistic reachable ranges. See also Subsec. 3.2.
7. `Get_Preference_From_DM` stores the most recent DM's preferences as \mathbf{z}^{pref} .
8. `Compute_Next_Iteration_Point` calculates the step point for the next step as described in (9).
9. `Max_Expected_ASF` uses the surrogate models s_1, \dots, s_k to find a solution that follows the DM's preferences and is likely to lie close to the Pareto front, as described in Sect. 3.4. This solution is then evaluated using the exact objective functions and added to the set of known solutions.
10. `Display_Chosen_Solutions` displays the solutions chosen by the DM by calculating and plotting solutions from P that are reachable from $\mathbf{z}^{(i)}$.

A DM can affect the algorithm by providing preferences, controlling when and where to conduct exact function evaluations, and by terminating the algorithm once satisfied. The DM can also pause the algorithm at any point (for example, in between steps during navigation) e.g., to update preference information or to jump backwards. An analyst can further affect the algorithm by choosing the surrogate modeling algorithm used in `Train`, the optimization algorithms used in `Optimistic_Optimize` and `Individual_Optima`, and by setting the number of steps and the rate at which they progress.

4 Case study

In this section, we demonstrate how O-NAUTILUS can be used to solve a multiobjective optimization problem. The implementation of the O-NAUTILUS method and this example

are openly available at <https://desdeo.it.jyu.fi> as a part of the DESDEO software framework and in a Zenodo repository via the link <https://doi.org/10.5281/zenodo.5396677>. Kriging is used as the surrogate modeling technique and RVEA [5] as the evolutionary multiobjective optimization algorithm to find the optimistic front as it generalizes well to a high number of objectives. For evaluating expected ASF, CMA-ES [13] is used in our implementation. CMA-ES is a state-of-the-art algorithm for continuous black-box optimization.

A video showcasing the UI of the O-NAUTILUS implementation can be viewed at <https://desdeo.it.jyu.fi/o-nautilus>. Data generated during the experiments can also be accessed via the same link.

4.1 Crash-worthiness design of vehicles

As the role of the DM is crucial in interactive methods, we can best demonstrate the applicability of these methods with problems where the objective functions are meaningful to a DM. Therefore, we apply O-NAUTILUS to a real-world engineering design optimization problem called crash-worthiness design of vehicles, originally proposed in [26]. It describes the design of the frontal structure of vehicles for crash safety optimization. When a car accident occurs, the frontal structure of the vehicle absorbs the energy caused by the crashing in order to increase the safety of the passengers. Improving the capacity of energy absorption can often lead to an increase in the total mass of the vehicle. However, lightweight designs are needed to reduce the mass and the fuel consumption of a vehicle, accordingly. Therefore, there is a trade-off between the safety and the environmental aspects, and a balanced decision must be made in the capacity of energy absorption and the mass of the vehicle.

In the problem design, a full frontal and an offset-frontal crash test are considered to simulate real-world accidents. A full-frontal crash usually results in a higher deceleration compared to an offset-frontal crash, which can cause severe injuries to the passengers. An offset-frontal test is needed to assess the structural integrity of a vehicle. In the design of the frontal structure, these two crashing types should be considered simultaneously to improve the overall crash-worthiness of the vehicle. Hence, there are three objectives: minimizing the mass of a vehicle (f_1 , in kg), minimizing deceleration during the full-frontal crash (f_2 , in m/s) and minimizing toe-board intrusion in the offset-frontal crash (f_3 , in m). The thickness of five components of the frontal structure of the vehicle have been chosen as the decision variables. A mathematical formulation of the multiobjective optimization problem can be found in Appendix A (see [26] for more details).

4.2 Interactive solution process

Next, we describe the interactive solution process using the O-NAUTILUS method. Before involving the DM, 100 sampling points (P) were generated using the latin hypercube sampling (LHS) technique based on the objective functions. The default number of steps for the navigation was set as 100 and the default rate of navigation as 10 steps per second. First, Kriging models were trained with the available sampling points. We then evaluated the optimistic front (P^+) by using RVEA on the trained Kriging models, as described in Sect. 3.1.

The combined set utopian and nadir points were calculated from P and P^+ as (1661.58, 6.46, 0.058) and (1693.61, 11.28, 0.23), respectively. Initially, the known and optimistic reachable ranges were calculated, as described in Sect. 3.2. The reachable ranges were then shown to the DM in a graphical user interface.

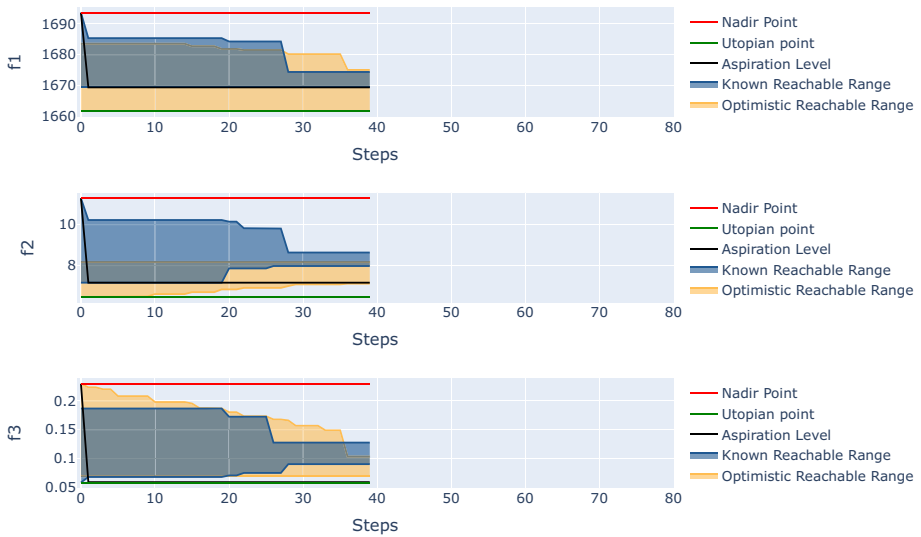


Fig. 7 Navigation view until the first change in the preference information

First, the DM wanted to provide aspiration levels for each objective based on the lower bounds of the known reachable range. As these values seemed very promising, the DM set the aspiration levels 1669.39 kg for f_1 , 7.16 m/s for f_2 and 0.058 m for f_3 , close to the combined set utopian point. Then, the navigation process was started to see whether these values were achievable or not. The DM saw real-time movement, in the sense that the reachable region started shrinking. Figure 7 shows the navigator view in the user interface of O-NAUTILUS. In this view, the combined set utopian and nadir point is shown by green and red lines, respectively. Also, the aspiration level provided by the DM is shown by the black line. As the navigation continues, the known and the optimistic reachable ranges are shown in blue (darker) and orange (lighter) areas, respectively. As can be seen in Fig. 7, the DM stopped the navigation since the aspiration levels for f_2 and f_3 became unachievable based on the known reachable ranges (shown in blue (darker shade)). However, the optimistic reachable ranges (shown in orange (lighter shade)) indicated that solutions close to the aspiration levels may still be achievable. Therefore, the DM decided to evaluate a new point and provided aspiration levels for f_2 and f_3 based on the current lower bounds of the optimistic reachable ranges with the intention to find a solution in the orange (lighter) area. The new reference point was (1669.39, 7.09, 0.07) and one additional function evaluation was made by using this reference point and eASF, as described in Sect. 3.4. The newly evaluated point (1672.33, 7.30, 0.08) had better objective function values than the current lower bounds of the known reachable area and was added to the set of known solutions.

The navigation was restarted with the preference information which was provided for the additional function evaluation. After a few steps, the known reachable area started to shrink and as a consequence of the newly added point, the DM realized that the aspiration levels for f_2 and f_3 were near the new lower bounds of the known reachable area. Based on the optimistic reachable ranges, there could be solutions better in the first objective without sacrificing in others. Therefore, the DM stopped the navigation and wanted to make another function evaluation by providing a new reference point (1661.58, 7.09, 0.07). The newly

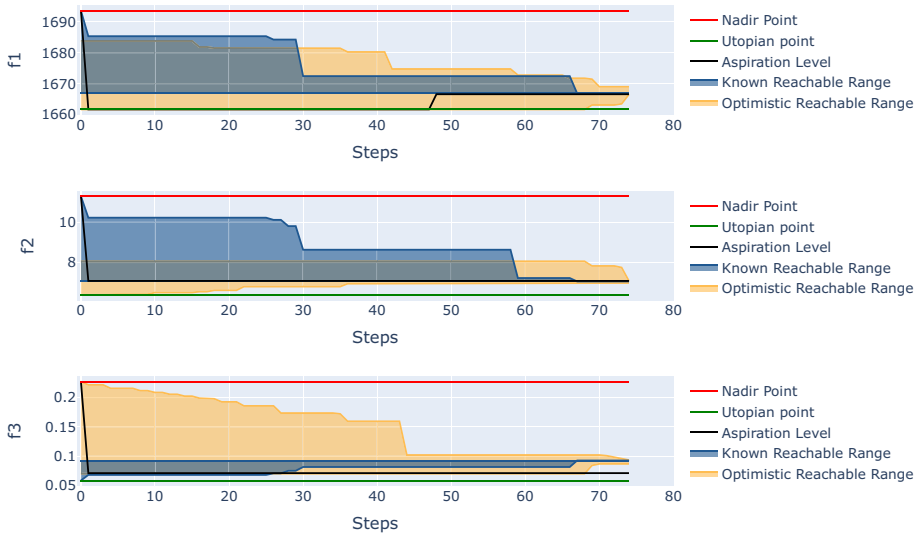


Fig. 8 Navigation view until the Pareto front is reached

evaluated point (1666.37, 7.05, 0.09) had better values than the current lower bound of the known reachable area for the first two objectives and was added to the set of known solutions.

The DM restarted the navigation with the newly added point. After a few steps of the navigation, the DM realized that the known reachable areas were shrinking and while the aspiration levels for f_2 and f_3 were achievable, the aspirations for f_1 was not. Then, he decided to pause the navigation in order to adjust his aspiration level for the first objective based on the lower bounds of known reachable area. He provided a new reference point (1666.60, 7.09, 0.07), which indicates a relaxation in the first objective. At this moment, he decided to let the navigation reach the Pareto front to see if his desires were achievable or not. The change in the aspiration levels and the navigation reaching the Pareto front are illustrated in Fig. 8.

As shown in Fig. 8, the reached solution reflected the desired values for the first and second objectives but not for the third objective. Because of this, the DM was not fully satisfied and realized that it may be possible to improve the third objective based on the final lower bound of the optimistic reachable ranges. Therefore, the DM wanted to conduct one more function evaluation by adjusting the aspiration level for the third objective according to the current optimistic front. He kept the same aspiration levels for the first two objectives. The new reference point for additional function evaluation was (1664.60, 7.09, 0.07). Then, the navigation was restarted with this reference point and the newly found solution added to the known set. The DM did not change his preference information in this last navigation since he had gained enough insight about the feasibility of his aspiration levels. Thus, he let the navigation reach a new final solution. As can be seen in Fig. 9, this time, the reached solution (1667.53, 7.22, 0.08) reflected the DM's preference information quite well, and the DM selected it as the most preferred solution. Table 1 summarizes the details of the interactive solution process. The table lists the values of combined set nadir and utopian point, lower and upper bounds of the known and optimistic reachable ranges, and the aspiration levels given for each objectives for each iterations. By iterations, we mean instances when the DM paused the navigation and provided preference information.

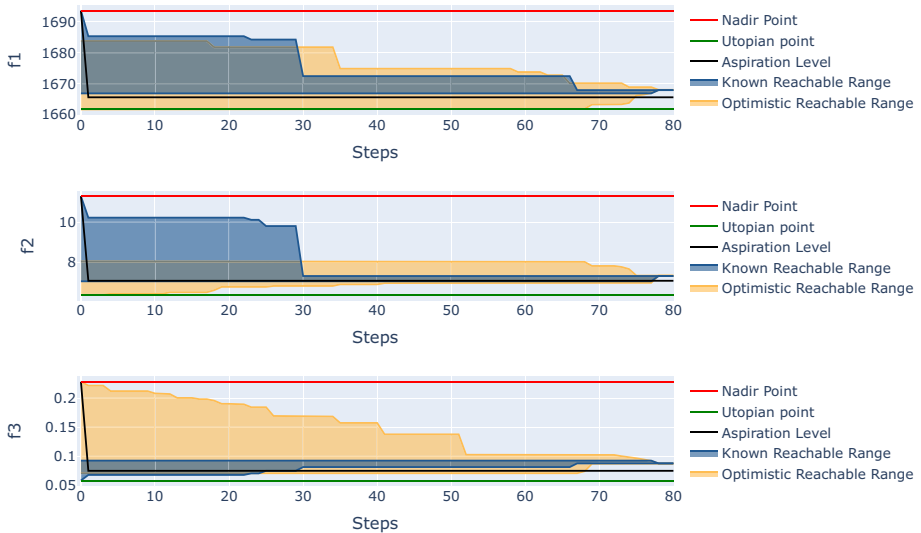


Fig. 9 Navigation view until a satisfactory solution was reached

4.3 Comparison with NAUTILUS navigator

To the best of our knowledge, no quality indicators for assessing and comparing the performance of interactive methods have been developed yet in the literature [1]. Therefore, we make the comparison by solving the same problem with NAUTILUS Navigator.

With O-NAUTILUS, the DM made only 3 additional exact objective function evaluations to reach the most preferred solution. As pointed out in the previous section, 100 function evaluations were made to generate LHS sampling data as an input for the O-NAUTILUS method. Therefore, a total of 103 function evaluations were conducted. Since in O-NAUTILUS, Kriging was used as a surrogate modeling technique and RVEA as an evolutionary algorithm, we used K-RVEA [6], an extension of RVEA which uses Kriging models, to generate the initial data set needed by NAUTILUS Navigator. K-RVEA generated 20 nondominated solutions with the same budget (103 exact function evaluations; 53 for LHS sampling data and 50 for K-RVEA). K-RVEA was run for the same number of generations as RVEA in the previous experiment.

With NAUTILUS Navigator, the DM provided similar preference information per iteration and eventually reached a satisfactory solution, which was (1668.49, 7.00362, 0.13276), in a similar manner. When this solution is compared with the one obtained by O-NAUTILUS, the DM reached a solution reflecting his preferences for the first two objectives but not for the last one. Therefore, we can say that he sacrificed more in the third objective in comparison to the final solution reached with O-NAUTILUS. Based on the DM’s experience with both methods, for the sake of brevity we list the differences without numerical details below:

- Optimistic ranges of O-NAUTILUS supported the DM in providing preference information and finding a region of interest.
- Additional function evaluations of O-NAUTILUS supported the DM in finding and adding new solutions in the region of interest which remained unexplored in NAUTILUS Navigator.

Table 1 Summary of some data shown to and provided by the DM when the DM paused the navigation and provided aspiration levels

Iteration number	Objectives	Nadir point	Ideal point	Known reachable		Optimistic reachable		Aspiration levels
				Max	Min	Max	Min	
1	f_1	1693.61	1661.58	1693.61	1669.39	1682.59	1661.58	1669.39
	f_2	11.28762	6.46597	11.28762	7.16433	8.08408	6.46597	7.16433
	f_3	0.23015	0.05833	0.18663	0.05833	0.23015	0.07099	0.05833
2	f_1	1693.61	1661.58	1674.39	1669.39	1673.39	1661.58	1669.39
	f_2	11.28762	6.46595	8.63587	7.97711	8.08402	7.09049	7.09049
	f_3	0.23015	0.05833	0.12756	0.09021	0.14123	0.07099	0.07099
3	f_1	1693.61	1661.58	1672.34	1669.39	1675.29	1661.58	1661.58
	f_2	11.28762	6.43311	8.63587	7.97711	8.08625	7.00589	7.09049
	f_3	0.22751	0.05833	0.12756	0.09021	0.10212	0.07080	0.07099
4	f_1	1693.61	1661.58	1672.34	1669.39	1675.29	1661.58	1666.60
	f_2	11.28762	6.43311	8.63587	7.97711	8.08625	7.00589	7.09049
	f_3	0.22751	0.05833	0.12756	0.09021	0.10212	0.07080	0.07099
5	f_1	1693.61	1661.68	1666.60	1666.60	1668.76	1666.51	1666.60
	f_2	11.28762	6.39665	6.99398	6.99398	7.47180	6.98596	7.09049
	f_3	0.22684	0.05833	0.09337	0.09337	0.09397	0.08507	0.08507

- NAUTILUS Navigator needed optimized points to start the solution process. Because of this, there is a pre-processing stage to generate nondominated solutions by using some appropriate multiobjective optimization method, for instance an evolutionary algorithm. On the contrary, O-NAUTILUS can start with any initial data which is not optimized. This means that if the DM has some available data of function evaluations for her/his optimization problem, it can readily be utilized.
- In NAUTILUS Navigator, the satisfaction degree of the final solution was highly dependent on the nondominated solutions found in the pre-processing stage. However, in O-NAUTILUS, even if initial solutions are not good, the DM can reach very good solutions with the help of additional function evaluations during the solution process. They will be conducted in a focused manner based on her/his preference information.

The positive features of O-NAUTILUS have some trade-offs as well. Because of the optimistic ranges and ability to make additional function evaluations, the DM needed to digest more information at a time in O-NAUTILUS. In NAUTILUS Navigator, as the DM saw only the reachable ranges, and no additional function evaluations were made in the solution process, NAUTILUS Navigator was easier to use than the O-NAUTILUS method. Furthermore, in O-NAUTILUS, the DM needs to wait for the additional exact function evaluation during the solution process. In NAUTILUS Navigator, waiting is out of the question during the interactive phase since no additional function evaluations are conducted during the solution process.

To conclude the comparison, O-NAUTILUS effectively supported the DM by providing optimistic reachable ranges and enabling additional function evaluations in the region of interest of the DM. Even with randomly generated initial points, the method could achieve satisfactory solutions by using few exact function evaluations. However, one needs to digest more information in each iteration and wait for the function evaluations conducted in the solution process.

5 Conclusions

In this paper we have proposed a novel method in the NAUTILUS family, termed O-NAUTILUS for interactive multiobjective optimization. The overarching challenge was to propose a new version of the NAUTILUS family that can be used for optimization problems where expensive function evaluations are to be used sparingly. To meet this challenge, we had to design a new, general algorithmic framework by integrating surrogate models with uncertainty handling into NAUTILUS methods. The new methodology allows for an interactive mode of exploration with alternating phases of decision making and compute-intensive steps where new evaluations of expensive objective functions are conducted to acquire information about promising regions of the decision space.

We have introduced novel concepts to tackle problems with expensive function evaluations. These new concepts have been implemented as an interactive O-NAUTILUS algorithm supported by a graphical user interface. With this user interface, the decision maker can see in real time how the preferences provided affect the direction of the search in terms of reachable ranges evolving. The usefulness of the new concepts is demonstrated as follows:

- We developed O-NAUTILUS as an extension of NAUTILUS Navigator by showing optimistic ranges to the decision maker. This allows the decision maker to schedule additional exact function evaluations to explore regions of the objective space where

(s)he can still expect (based on optimistic bounds) further improvements. This is useful for optimization with expensive function evaluations.

- We created an open source implementation of O-NAUTILUS in Python which can be accessed at <https://desdeo.it.jyu.fi>.
- We demonstrated the implementation on an instance from the domain of crashworthiness design, which is a typical context where expensive function evaluations occur in practice.
- We augmented the graphical representation of the reachable ranges plot to accommodate optimistic ranges.
- We made several detailed decisions when developing the ideas into a workable algorithm: the type of user interaction and preference model and the choice of surrogate modeling and uncertainty handling techniques. Kriging models were used but also other surrogate models providing uncertainty quantification can be utilized. As an example of alternative models, we mentioned Lipschitzian models.

In this contribution, we have introduced surrogate modeling techniques with uncertainty handling for the first time in the context of NAUTILUS methods. Surrogate modeling is in itself a very active field of research which has recently brought forward many results. Future work should therefore include the choice and update of the surrogate modeling techniques. One should note that although Kriging and Lipschitzian models are utilized in this paper to approximate the optimistic Pareto front, the ideas are not limited to these two surrogate model types and one can also use other surrogate model types which feature uncertainty quantification. Understanding when to apply Kriging and when Lipschitzian models, or other types of surrogate models deserves further analysis. Further experiments are also to be conducted with O-NAUTILUS, especially with real-life problems.

Acknowledgements This research was partly funded by the Academy of Finland (Grants 322221 and 311877). The research is related to the thematic research area Decision Analytics utilizing Causal Models and Multi-objective Optimization (DEMO), jyu.fi/demo, at the University of Jyväskylä.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

A Mathematical formulation of the multiobjective optimization problem for the crash-worthiness design of vehicle

Following [26], the multiobjective optimization problem for the crash safety design of vehicles is formulated as follows:

$$\begin{aligned} & \text{minimize } F(\mathbf{x}) = \{f_1(\mathbf{x}), f_2(\mathbf{x}), f_3(\mathbf{x})\} \\ & \text{subject to } 1 \leq x_j \leq 3 \quad j = 1, \dots, 5, \end{aligned} \quad (11)$$

where f_i ($i = 1, 2, 3$) representing the relevant surrogate models, formed based on the data collected from simulation models as described in [26], have the following formulations:

$$\begin{aligned} f_1(\mathbf{x}) &= 1640.2823 + 2.3573285x_1 + 2.3220035x_2 + 4.5688768x_3 \\ &\quad + 7.7213633x_4 + 4.4559504x_5 \\ f_2(\mathbf{x}) &= 6.5856 + 1.15x_1 - 1.0427x_2 + 0.9738x_3 + 0.8364x_4 \\ &\quad - 0.3695x_1x_4 + 0.0861x_1x_5 + 0.3628x_2x_4 \\ &\quad - 0.1106x_1^2 - 0.3437x_3^2 + 0.1764x_4^2 \\ f_3(\mathbf{x}) &= -0.0551 + 0.0181x_1 + 0.1024x_2 + 0.0421x_3 - 0.0073x_1x_2 \\ &\quad + 0.024x_2x_3 - 0.0118x_2x_4 - 0.0204x_3x_4 - 0.008x_3x_5 \\ &\quad - 0.0241x_2^2 + 0.0109x_4^2 \end{aligned}$$

Here, the objective functions f_1 , f_2 , and f_3 are the vehicle mass, deceleration during the full-frontal crash, and toe-board intrusion in the offset-frontal crash, respectively. The decision variable x_j ($j = 1, \dots, 5$) represents the thickness of a component of the frontal structure of the vehicle. See [26] for more details about the surrogate construction and problem formulation.

References

1. Afsar, B., Miettinen, K., Ruiz, F.: Assessing the performance of interactive multiobjective optimization methods: a survey. *ACM Comput. Surv.* **54**(4), 85 (2021)
2. Audet, C.: A Survey on Direct Search Methods for Blackbox Optimization and Their Applications. In: Pardalos, P.M., Rassias, T.M. (eds.) *Mathematics without boundaries*, vol. 2, pp. 31–56. Springer, Berlin (2014)
3. Aytuğ, H., Sayın, S.: Using support vector machines to learn the efficient set in multiple objective discrete optimization. *Eur. J. Oper. Res.* **193**(2), 510–519 (2009)
4. Buchanan, J.T., Corner, J.: The effects of anchoring in interactive MCDM solution methods. *Comput. Oper. Res.* **24**(10), 907–918 (1997)
5. Cheng, R., Jin, Y., Olhofer, M., Sendhoff, B.: A reference vector guided evolutionary algorithm for many-objective optimization. *IEEE Trans. Evol. Comput.* **20**(5), 773–791 (2016)
6. Chugh, T., Jin, Y., Miettinen, K., Hakanen, J., Sindhya, K.: A surrogate-assisted reference vector guided evolutionary algorithm for computationally expensive many-objective optimization. *IEEE Trans. Evol. Comput.* **22**(1), 129–142 (2016)
7. Chugh, T., Sindhya, K., Hakanen, J., Miettinen, K.: A survey on handling computationally expensive multiobjective optimization problems with evolutionary algorithms. *Soft Comput.* **23**(9), 3137–3166 (2019)
8. Deb, K.: *Multi-objective Optimization Using Evolutionary Algorithms*. Wiley (2001)
9. Emmerich, M.: Single- and multi-objective evolutionary design optimization assisted by Gaussian random field metamodelling. PhD dissertation, University of Dortmund (2005)
10. Emmerich, M., Yang, K., Deutz, A., Wang, H., Fonseca, C.M.: A Multicriteria Generalization of Bayesian Global Optimization. In: Pardalos, P.M., Zhigljavsky, A., Žilinskas, J. (eds.) *Advances in stochastic and deterministic global optimization*, pp. 229–242. Springer, Berlin (2016)
11. Floudas, C.A., Pardalos, P.M. (eds.): *Encyclopedia of Optimization*. Springer, Berlin (2008)
12. Gaudrie, D., Le Riche, R., Picheny, V., Enaux, B., Herbert, V.: Targeting solutions in Bayesian multi-objective optimization: sequential and batch versions. *Ann. Math. Artif. Intell.* **88**(1), 187–212 (2020)
13. Hansen, N., Ostermeier, A.: Completely derandomized self-adaptation in evolution strategies. *Evol. Comput.* **9**(2), 159–195 (2001)
14. Hartikainen, M., Miettinen, K., Klamroth, K.: Interactive nonconvex pareto navigator for multiobjective optimization. *Eur. J. Oper. Res.* **275**(1), 238–251 (2019)
15. Hastings, W.D.: Monte Carlo sampling methods using Markov chains and their applications. *Biometrika* **57**(1), 97–109 (1970)
16. Horst, R., Pardalos, P.M. (eds.): *Handbook of Global Optimization*. Springer, Berlin (1995)

17. Husain, A., Kim, K.-Y.: Enhanced multi-objective optimization of a microchannel heat sink through evolutionary algorithm coupled with multiple surrogate models. *Appl. Therm. Eng.* **30**(13), 1683–1691 (2010)
18. Jin, Y., Wang, H., Chugh, T., Guo, D., Miettinen, K.: Data-driven evolutionary optimization: an overview and case studies. *IEEE Trans. Evol. Comput.* **23**(3), 442–458 (2019)
19. Jones, D.R.: A taxonomy of global optimization methods based on response surfaces. *J. Glob. Opt.* **21**(4), 345–383 (2001)
20. Jones, D.R., Schonlau, M., Welch, W.J.: Efficient global optimization of expensive black-box functions. *J. Glob. Opt.* **13**(4), 455–492 (1998)
21. Kahneman, D., Tversky, A.: Prospect theory: an analysis of decisions under risk. *Econometrica* **47**, 313–327 (1979)
22. Knowles, J.: ParEGO: a hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Trans. Evol. Comput.* **10**(1), 50–66 (2006)
23. Kourakos, G., Mantoglou, A.: Development of a multi-objective optimization algorithm using surrogate models for coastal aquifer management. *J. Hydrol.* **479**, 13–23 (2013)
24. Kvasov, D.E., Sergeyev, Y.D.: Lipschitz global optimization methods in control problems. *Autom. Remote Control* **74**(9), 1435–1448 (2013)
25. Li, M., Li, G., Azarm, S.: A Kriging metamodel assisted multi-objective genetic algorithm for design optimization. *J. Mech. Des.* **130**(3), 031401 (2008)
26. Liao, X., Li, Q., Yang, X., Zhang, W., Li, W.: Multiobjective optimization for crash safety design of vehicles using stepwise regression model. *Struct. Multidiscipl. Opt.* **35**(6), 561–569 (2008)
27. Miettinen, K.: *Nonlinear Multiobjective Optimization*. Kluwer Academic Publishers, Boston (1999)
28. Miettinen, K., Eskelinen, P., Ruiz, F., Luque, M.: NAUTILUS method: an interactive technique in multi-objective optimization based on the nadir point. *Eur. J. Oper. Res.* **206**(2), 426–434 (2010)
29. Miettinen, K., Hakanen, J., Podkopaev, D.: *Interactive Nonlinear Multiobjective Optimization Methods*. In: Greco, S., Ehrgott, M., Figueira, J. (eds.) *Multiple criteria decision analysis: state of the art surveys*, 2nd edn., pp. 931–980. Springer, Berlin (2016)
30. Miettinen, K., Ruiz, F.: NAUTILUS framework: towards trade-off-free interaction in multiobjective optimization. *J. Bus. Econ.* **86**(1–2), 5–21 (2016)
31. Miettinen, K., Ruiz, F., Wierzbicki, A.P.: Introduction to Multiobjective Optimization: Interactive Approaches. In: Branke, J., Deb, K., Miettinen, K., Slowinski, R. (eds.) *Multiobjective Optimization: Interactive and Evolutionary Approaches*, pp. 27–57. Springer, Berlin (2008)
32. Mitra, K., Majumder, S.: Successive approximate model based multi-objective optimization for an industrial straight grate iron ore induration process using evolutionary algorithm. *Chem. Eng. Sci.* **66**(15), 3471–3481 (2011)
33. Mockus, J., Tiesis, V., Zilinskas, A.: The application of Bayesian methods for seeking the extremum. *Towards Glob. Opt.* **2**(117–129), 2 (1978)
34. Paulavičius, R., Žilinskas, J., Grothey, A.: Investigation of selection strategies in branch and bound algorithm with simplicial partitions and combination of Lipschitz bounds. *Opt. Lett.* **4**(2), 173–183 (2010)
35. Pintér, J.D.: *Global Optimization in Action: Continuous and Lipschitz Optimization: Algorithms, Implementations and Applications*. Springer, Berlin (2013)
36. Qasem, S.N., Shamsuddin, S.M., Hashim, S.Z.M., Darus, M., Al-Shammari, E.: Memetic multiobjective particle swarm optimization-based radial basis function network for classification problems. *Inf. Sci.* **239**, 165–190 (2013)
37. Regis, R.G., Shoemaker, C.A.: Combining radial basis function surrogates and dynamic coordinate search in high-dimensional expensive black-box optimization. *Eng. Opt.* **45**(5), 529–555 (2013)
38. Ruiz, A.B., Ruiz, F., Miettinen, K., Delgado-Antequera, L., Ojalehto, V.: NAUTILUS Navigator: free search interactive multiobjective optimization without trading-off. *J. Glob. Opt.* **74**(2), 213–231 (2019)
39. Sergeyev, Y.D., Kvasov, D.E.: *Deterministic Global Optimization: An Introduction to the Diagonal Approach*. Springer, Berlin (2017)
40. Sergeyev, Y.D., Kvasov, D.E., Mukhametzanov, M.S.: On the efficiency of nature-inspired metaheuristics in expensive global optimization with limited budget. *Sci. Rep.* **8**(453), 1–9 (2018)
41. Stein, M.L.: *Interpolation of Spatial Data: Some Theory for Kriging*. Springer, New York, NY (1999)
42. Strongin, R.G., Sergeyev, Y.D.: *Global Optimization with Non-convex Constraints: Sequential and Parallel Algorithms*. Springer, Berlin (2013)
43. Tabatabaei, M., Hakanen, J., Hartikainen, M., Miettinen, K., Sindhya, K.: A survey on handling computationally expensive multiobjective optimization problems using surrogates: non-nature inspired methods. *Struct. Multidiscip. Opt.* **52**(1), 1–25 (2015)
44. Van Beers, W.C.M., Kleijnen, J.P.C.: Kriging for interpolation in random simulation. *J. Oper. Res. Soc.* **54**(3), 255–262 (2003)

45. Wierzbicki, A.P.: On the completeness and constructiveness of parametric characterizations to vector optimization problems. *OR Spektrum* **8**, 73–87 (1986)
46. Zhigljavsky, A., Žilinskas, A.: *Stochastic Global Optimization*, vol. 9. Springer Science & Business Media, Berlin (2007)
47. Žilinskas, A.: Visualization of a statistical approximation of the Pareto front. *Appl. Math. Comput.* **271**, 694–700 (2015)
48. Žilinskas, A., Žilinskas, J.: Adaptation of a one-step worst-case optimal univariate algorithm of bi-objective Lipschitz optimization to multidimensional problems. *Commun. Nonlinear Sci. Numer. Simul.* **21**(1–3), 89–98 (2015)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.