Surrogate-Assisted Bounding-Box Approach for Optimization Problems with Tunable Objectives Fidelity

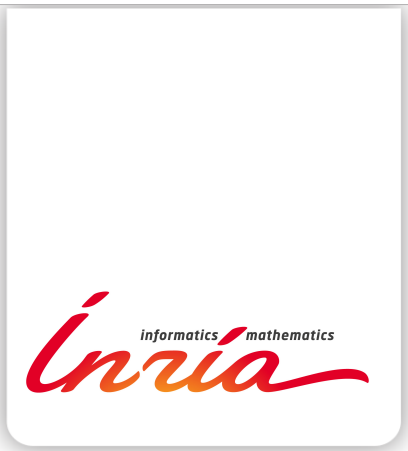
Mickael Rivier, Pietro Marco Congedo

Surrogate-Assisted Bounding-Box Approach for Optimization Problems with Tunable Objectives Fidelity

Mickaël Rivier, Pietro Marco Congedo

RESEARCH REPORT
N° 9155
September 12, 2019
Project-Teams DEFI

ISSN 0249-6399 ISRN INRIA/RR--9155--FR+ENG



Surrogate-Assisted Bounding-Box Approach for Optimization Problems with Tunable Objectives Fidelity

Mickaël Rivier*†, Pietro Marco Congedo *

Project-Teams DEFI

Research Report n° 9155 — September 12, 2019 — 32 pages

Abstract: In this work, we present a novel framework to perform multi-objective optimization when considering expensive objective functions computed with tunable fidelity. This case is typical in many engineering optimization problems, for example with simulators relying on Monte Carlo or on iterative solvers. The objectives can only be estimated, with an accuracy depending on the computational resources allocated by the user. We propose here a heuristic for allocating the resources efficiently to recover an accurate Pareto front at low computational cost. The approach is independent from the choice of the optimizer and overall very flexible for the user.
The framework is based on the concept of Bounding-Box, where the estimation error can be regarded with the abstraction of an interval (in one-dimensional problems) or a product of intervals (in multi-dimensional problems) around the estimated value, naturally allowing the computation of an approximated Pareto front. This approach is then supplemented by the construction of a surrogate model on the estimated objective values.
We first study the convergence of the approximated Pareto Front toward the true continuous one under some hypotheses. Secondly, a numerical algorithm is proposed and tested on several numerical test-cases.

Key-words: Multi-objective optimization, Uncertainty-based optimization, Error Bounding-Boxes, Tunable fidelity, Surrogate-assisting strategy

* DeFI - CMAP - Ecole Polytechnique, Inria Saclay - Ile de France, Polytechnique - X, CNRS
† ArianeGroup, Le Haillan

RESEARCH CENTRE
SACLAY – ÎLE-DE-FRANCE

1 rue Honoré d'Estienne d'Orves
Bâtiment Alan Turing
Campus de l'École Polytechnique
91120 Palaiseau

Le framework SABBa pour les problèmes d'optimisation avec des objectifs à fidélité adaptable

Résumé : Dans ce travail, nous présentons un nouveau cadre pour la résolution de problèmes d'optimisation multi-objectif en considérant que le calcul de ces objectifs peut se faire avec une fidélité adaptable, et que la haute fidélité est extrêmement coûteuse. Ce genre de problème est souvent rencontré en ingénierie, avec des fonctions objectifs reposant sur des calculs Monte Carlo ou des solveurs itératifs. Ces objectifs ne peuvent être qu'estimés, avec une précision dépendant fortement des ressources allouées par l'utilisateur. Nous proposons ici une heuristique permettant d'allouer efficacement les ressources de calculafin d'obtenir une front de Pareto précis à moindre coût de calcul. Cette approche ne dépend pas du choix de l'algorithme d'optimisation et laisse une grande flexibilité à l'utilisateur.

La strategie proposée est basée sur le concept de Boîtes conservatives, où l'erreur d'estimation est représentée par un intervalle (pour les problèmes mono-objectif) ou un produit d'intervalles (pour les problèmes multi-objectif) autour de la valeur estimée. Cela permet naturellement la création d'un front de Pareto approximé. Cette approche est de plus couplée à la construction d'un modèle de substitution sur les objectifs différents objectifs.

La convergence du front de Pareto approximé vers le front réel est d'abord étudiée sous quelques hypothèses. Un algorithme est ensuite proposé et testé sur plusieurs cas-tests numériques.

Mots-clés : Optimisation multi-objectif, Optimisation sous incertitudes, Boîtes d'erreur conservatives, Fidélité adaptable, Assistance par modèle de substitution

1 Introduction

A key ingredient in any optimization method is the evaluation of the objective functions, which provides the metric to discriminate a good and feasible design, according to a set of objectives and constraints. In many real applications, the evaluation of the objective functions can be affected by a source of uncertainty, noise or error. A possible classification is presented in [1], where four classes are identified: (i) noise, caused for example by sensor measurements or random simulations, (ii) uncertainty for robustness (or reliability) analysis, where the model/numerical method used to compute the objective functions includes stochastic variables, (iii) approximation error due to an approximate model of the exact objective functions and (iv) time-varying functions.

We focus here on a type of approximation error, which can, for example, arise in iterative or sampling-based computations. The exact objective values are the limits, which are estimated at a tunable cost and fidelity. A tunable fidelity does not mean that the user chooses a priori the fidelity of each objective, but rather that there is a way of increasing the fidelity of the black-box computation at a given design. We consider all objectives to result from a single expensive solver, all objectives are then refined simultaneously each time the fidelity is increased. Ref. [2] introduced the idea of tunable fidelity and proposed an automated allocation strategy for mono-objective kriging-based optimizations.

The scope of the study presented here is to formulate a framework permitting to solve a multi-objective optimization efficiently with tunable objectives fidelity, independently from the choice of the optimizer. We target optimization problems linked to the resolution of an iterative solver. A non-exhaustive list includes the Gauss-Seidel method, Newton method, segregated approach, additive Schwarz technique, or computation of a sampling-based objective. The later notably covers uncertainty-based optimization, where the objectives are statistics approximated with Monte Carlo approaches, and more generally any optimization of expensive integrands [3]. Note that the use of the concept of Pareto dominance limits the applicability of the proposed method to multi-objective optimization problems with two-three objectives. Many-objectives problems (more than three objectives) require dedicated techniques to discriminate between all non-dominated designs (see *e.g.* [4]).

Several techniques have been developed to cope with noise in objective values in the context of stochastic search algorithms. Review papers [1, 5] depicted the main methods to deal with uncertainties in an Evolutionary Algorithm (EA) environment, namely averaging, robust index selection or robustness measures. A robustness measure can be seen as a new fitness assessment, as in [6], or as an additional robustness index objective in [7]. Deb et al [8] introduced two concepts of robustness, revising the objective function or applying an additional constraint on the problem. The main strategy to account for uncertainty on the objective functions is to use a probabilistic ranking of the individuals. The Stochastic Pareto Genetic Algorithm, developed in [9], computes probabilistic dominations between individual, assuming normal distributions, and uses them for ranking. Hughes [10] also assumes normal distributions and proposes a probabilistic extension of the Non-dominated Sorting Genetic Algorithm ranking. This work was then supplemented by [11] with a Bayesian learning of the variances of input noises. Teich [12] followed an idea similar to [10] but with uniform distributions of the objective values. Other heuristics like Simulated Annealing and Pure Random Search have been developed in this context in [13] and [14]. Model-based approaches have been applied to noisy optimization problems in [15]. Non-probabilistic interval uncertainties have been tackled through new dominance criteria in [16, 17], or with a worst case methodology in [18]. Epistemic uncertainties have also been treated by means of evidence theory in [19, 20].

In the review of Mlakar et al. [21], several techniques of optimization with approximated

values have been compared. The authors introduce the concept of Bounding-Boxes, with an error regarded with the abstraction of an interval (in one-dimensional problems) or a product of intervals (in multi-dimensional problems) around the estimated value. They also introduce an extended concept of the Pareto dominance rule. In Fusi et al. [22], these concepts have been used for online resource allocation in the context of robust and reliability-based optimization, where the objective functions are statistics of a quantity of interest. The Bounding-Box approach allows to tune the fidelity of the statistics computation depending on the closeness to the Pareto front.

The first contribution of this paper relies on providing some mathematical evidence about the convergence of the Bounding-Box approach for multi-objective optimization problem with adaptively tuned objective values. The work [22] revealed that while substantial gains can be obtained in the early stages of the optimization, the convergence of the optimizer toward the Pareto front lessen the efficiency of the Bounding-Box approach. This behavior is due to the increasing number of efficient designs when the optimizer finds the optimal area.

For curing this issue, as a second contribution, we couple the Bounding-Box approach with a Surrogate-Assisting strategy. This should further reduce the computational cost, notably during the last iterations of the optimization process. In practice, the Bounding-Box approach can quickly drive the optimizer toward the Pareto optimal area, where the Surrogate-Assisting model could permit to increase the accuracy locally. Such acceleration strategies using surrogate models have been studied in the optimization context in [23], and notably within the Evolutionary Strategies ([24, 25, 26, 27, 28]).

The aim of this paper is to introduce a framework called SABBa (Surrogate-Assisted Bounding-Box approach), and to provide mathematical and numerical evidence about its convergence properties. After explaining the strategy in Section 2, convergence of the Pareto front approximations is mathematically analyzed under several assumptions in Section 3. Then, the general pseudo-algorithm of the framework is presented in Section 4 and three analytical test-cases are fully analyzed in Section 5. Finally, conclusive remarks and future works are given in Section 6.

2 Overview of the SABBa framework

A multi-objective optimization problem can be stated as

$$\begin{aligned} \text{minimize/maximize:} \quad & \boldsymbol{f}(\boldsymbol{x}) \\ \text{by changing:} \quad & \boldsymbol{x} \in \mathcal{X} \end{aligned} \tag{1}$$

where the $m \leq 3$ objective functions are collected in a vector $\boldsymbol{f} \in \mathbb{R}^m$ and $\boldsymbol{x} \in \mathcal{X}$ are the n design variables included in the design domain $\mathcal{X} \subset \mathbb{R}^n$.

In this work, we assume that the objective functions $\boldsymbol{f}$ cannot be computed exactly, but only estimated. The associated error can be reduced through additional computational burden. These estimated objective values are denoted as $\widetilde{\boldsymbol{f}}^l(\boldsymbol{x})$ and associated to the error $\boldsymbol{\varepsilon}^l(\boldsymbol{x})$, so that $\widetilde{\boldsymbol{f}}^l(\boldsymbol{x}) = \boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{\varepsilon}^l(\boldsymbol{x})$. Here the tilda refers to the approximation intrinsic to the objectives computation, which accuracy depends on the level of fidelity l. This work aims at formulating an adaptive strategy for choosing an optimal l for each visited $\boldsymbol{x}$. Note then that l varies with respect to $\boldsymbol{x}$, *i.e.* $l(\boldsymbol{x})$. In the following, we decide to indicate the level of fidelity simply l and not $l(\boldsymbol{x})$ to shorten the notation.

This section illustrates the SABBa framework, based on the coupling between the Bounding-Box approach and the Surrogate-Assisting strategy. The Bounding-Box approach rely on the definitions of Bounding-Boxes and the associated Pareto dominance, given in Sections 2.1 and

2.2, respectively. This approach and the Surrogate-Assisting strategy are then described in Section 2.3 and 2.4. The flowchart of the framework is also illustrated.

2.1 Definition of a Bounding-Box in a multi-objective problem

As stated above, the objective functions are intrinsically subject to an error $\boldsymbol{\varepsilon}^l(\boldsymbol{x})$, correlated with the level of fidelity l. We denote with $\widetilde{\boldsymbol{\varepsilon}}^l(\boldsymbol{x})$ a conservative estimation of this error, so that $|\boldsymbol{\varepsilon}^l(\boldsymbol{x})| \leq \widetilde{\boldsymbol{\varepsilon}}^l(\boldsymbol{x})$, *i.e.* $\boldsymbol{f} \in \left[\widetilde{\boldsymbol{f}}^l - \widetilde{\boldsymbol{\varepsilon}}^l, \widetilde{\boldsymbol{f}}^l + \widetilde{\boldsymbol{\varepsilon}}^l\right]$. This m-dimensional interval containing $\boldsymbol{f}$ (represented in Figure 1) is called a *Box*. It is denoted as $\mathcal{B}\big(\widetilde{\boldsymbol{f}}^l, \widetilde{\boldsymbol{\varepsilon}}^l\big)$ using the notation presented in the following Definition 1.

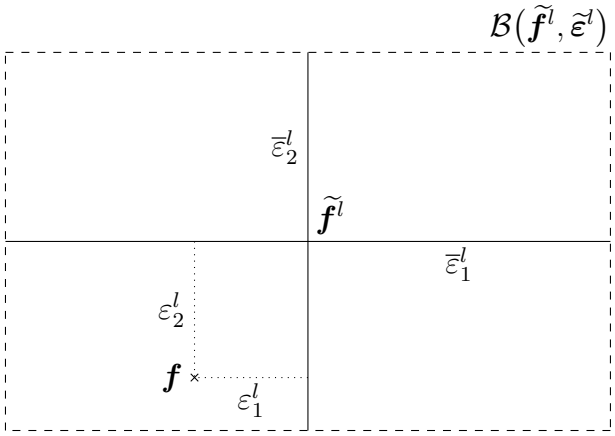


Figure 1: Bounding-Box approximation

Definition 1. A m-dimensional box is defined by its center and half-width vectors as follows:

$$\mathcal{B}(\boldsymbol{a}, \boldsymbol{r}) = \big\{\boldsymbol{b} \in \mathbb{R}^m \mid \boldsymbol{b} \in [\boldsymbol{a} - \boldsymbol{r}, \boldsymbol{a} + \boldsymbol{r}]\big\} \in \wp(\mathbb{R}^m)$$

$\forall(\boldsymbol{a}, \boldsymbol{r}) \in \mathbb{R}^m \times \mathbb{R}_+^m$, and with $\wp(\mathbb{R}^m)$ the power set of $\mathbb{R}^m$.

Remarks

- $\mathcal{B}(\boldsymbol{a}, \boldsymbol{0}) \equiv \boldsymbol{a}$.
- To lighten the notation, with $\boldsymbol{f} : \mathcal{X} \to \mathbb{R}^m$, we set:

$$\mathcal{B}_{\boldsymbol{f}}(\boldsymbol{x}, \boldsymbol{r}) \equiv \mathcal{B}\big(\boldsymbol{f}(\boldsymbol{x}), \boldsymbol{r}\big), \quad \forall(\boldsymbol{x}, \boldsymbol{r}) \in \mathcal{X} \times \mathbb{R}^m.$$

2.2 Pareto dominance

In multi-objective optimization, objective functions are usually naturally conflicting, producing a set of trade-off optima. These designs are said Pareto-optimal and the ensemble is called the Pareto front. They are non-dominated according to the following Pareto dominance rule:

Definition 2. *Pareto dominance.* For two vectors $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^m$,

$$\begin{aligned}
\boldsymbol{a} \succ \boldsymbol{b} \ (\boldsymbol{a} \text{ dominates } \boldsymbol{b}) &\iff \forall j \in [\![1,m]\!],\ \pm a_j \leq \pm b_j \quad \text{and} \\
&\quad\ \exists j \in [\![1,m]\!],\ \pm a_j < \pm b_j, \\
\boldsymbol{a} \succ\!\!\succ \boldsymbol{b} \ (\boldsymbol{a} \text{ strictly dominates } \boldsymbol{b}) &\iff \forall j \in [\![1,m]\!],\ \pm a_j < \pm b_j, \\
\boldsymbol{a} \sim \boldsymbol{b} \ (\boldsymbol{a} \text{ is indifferent to } \boldsymbol{b}) &\iff \boldsymbol{a} \not\succ \boldsymbol{b} \text{ and } \boldsymbol{b} \not\succ \boldsymbol{a}.
\end{aligned}$$

$\forall j \in [\![1,m]\!]$, the symbol $\pm$ (implicitly $\pm_j$) indicates the goal in the j^{th} dimension:

$$\pm = \begin{cases} + & \text{for minimization,} \\ - & \text{for maximization.} \end{cases}$$

Definition 3. For two designs $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{X}$, where $\mathcal{X}$ is the design space, and $\boldsymbol{f}$ the objective functions, we introduce the following notations:

$$\begin{aligned}
\boldsymbol{x} \overset{\boldsymbol{f}}{\succ} \boldsymbol{y} &\iff \boldsymbol{f}(\boldsymbol{x}) \succ \boldsymbol{f}(\boldsymbol{y}), \\
\boldsymbol{x} \overset{\boldsymbol{f}}{\succ\!\!\succ} \boldsymbol{y} &\iff \boldsymbol{f}(\boldsymbol{x}) \succ\!\!\succ \boldsymbol{f}(\boldsymbol{y}), \\
\boldsymbol{x} \overset{\boldsymbol{f}}{\sim} \boldsymbol{y} &\iff \boldsymbol{f}(\boldsymbol{x}) \sim \boldsymbol{f}(\boldsymbol{y}).
\end{aligned}$$

Remark Note that the objective functions appear explicitly in the above notation.

Definition 4. A design $\boldsymbol{x} \in \mathcal{X}$ is said to be non-dominated (or efficient) regarding a set $\mathcal{A} \subseteq \mathcal{X}$ if and only if:

$$\nexists \boldsymbol{y} \in \mathcal{A},\ \boldsymbol{y} \overset{\boldsymbol{f}}{\succ} \boldsymbol{x}.$$

Then, $\boldsymbol{x}$ is said to be Pareto-optimal iff $\boldsymbol{x}$ is non-dominated regarding $\mathcal{X}$.

In this work, $\boldsymbol{f}$ is assumed to be unknown. Only the estimated value $\widetilde{\boldsymbol{f}}^l$ and its conservative error $\widetilde{\boldsymbol{\varepsilon}}^l$ can be computed, with $\boldsymbol{f} \in \mathcal{B}\big(\widetilde{\boldsymbol{f}}^l, \widetilde{\boldsymbol{\varepsilon}}^l\big)$. A new dominance criterion is then introduced.

Following [21] and [22], this dominance is denoted by *Boxed Pareto dominance*. Intuitively, a box $\mathcal{B}(\boldsymbol{a}, \boldsymbol{r})$ dominates another box $\mathcal{B}(\boldsymbol{b}, \boldsymbol{r}')$ if and only if any point of $\mathcal{B}(\boldsymbol{a}, \boldsymbol{r})$ dominates (in the classical way) every point of $\mathcal{B}(\boldsymbol{b}, \boldsymbol{r}')$. This can also be interpreted as a classical dominance rule between the least performing point of $\mathcal{B}(\boldsymbol{a}, \boldsymbol{r})$ and the most performing one of $\mathcal{B}(\boldsymbol{b}, \boldsymbol{r}')$.

Definition 5. *Boxed Pareto dominance.* For two boxes $\big(\mathcal{B}(\boldsymbol{a}, \boldsymbol{r}), \mathcal{B}(\boldsymbol{b}, \boldsymbol{r}')\big) \in \wp(\mathbb{R}^m)^2$,

$$\begin{aligned}
\mathcal{B}(\boldsymbol{a}, \boldsymbol{r}) \underset{\mathcal{B}}{\succ} \mathcal{B}(\boldsymbol{b}, \boldsymbol{r}') &\iff \forall j \in [\![1,m]\!],\ \pm a_j + r_j \leq \pm b_j - r'_j \quad \text{and} \\
&\quad\ \exists j \in [\![1,m]\!],\ \pm a_j + r_j < \pm b_j - r'_j, \\
\mathcal{B}(\boldsymbol{a}, \boldsymbol{r}) \underset{\mathcal{B}}{\succ\!\!\succ} \mathcal{B}(\boldsymbol{b}, \boldsymbol{r}') &\iff \forall j \in [\![1,m]\!],\ \pm a_j + r_j < \pm b_j - r'_j, \\
\mathcal{B}(\boldsymbol{a}, \boldsymbol{r}) \underset{\mathcal{B}}{\sim} \mathcal{B}(\boldsymbol{b}, \boldsymbol{r}') &\iff \mathcal{B}(\boldsymbol{a}, \boldsymbol{r}) \underset{\mathcal{B}}{\not\succ} \mathcal{B}(\boldsymbol{b}, \boldsymbol{r}') \text{ and } \mathcal{B}(\boldsymbol{b}, \boldsymbol{r}') \underset{\mathcal{B}}{\not\succ} \mathcal{B}(\boldsymbol{a}, \boldsymbol{r}).
\end{aligned}$$

The following equivalence is verified:

$$\forall \boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^m,\ \mathcal{B}(\boldsymbol{a}, 0) \underset{\mathcal{B}}{\succ} \mathcal{B}(\boldsymbol{b}, 0) \iff \boldsymbol{a} \succ \boldsymbol{b}, \tag{2}$$

which traduces the consistency between the two domination rules when the size of the box tends to zero.

Remark It can be noted that the relation between $\mathcal{B}(\boldsymbol{a}, \boldsymbol{r})$ and $\mathcal{B}(\boldsymbol{b}, \boldsymbol{r}')$ is the same as between $\mathcal{B}(\boldsymbol{a}, \boldsymbol{r}')$ and $\mathcal{B}(\boldsymbol{b}, \boldsymbol{r})$.

2.3 Bounding-Box approach

The Bounding-Box approach proposed in [22] is a strategy permitting to choose for each design a fidelity level l to compute the fitness function. It aims at optimizing the trade-off between accuracy and computational cost and relies on the *Boxed Pareto dominance* rule defined above. The steps are the following:

a) The chosen optimization algorithm provides a new design $\boldsymbol{x}$ (Note that there may be an ensemble of designs if using for example evolutionary algorithms);

b) The objectives are estimated at $\boldsymbol{x}$ with low-fidelity (l=0) to get the associated box $\mathcal{B}\big(\widetilde{\boldsymbol{f}}^0(\boldsymbol{x}), \widetilde{\boldsymbol{\varepsilon}}^0(\boldsymbol{x})\big)$;

c) This box is then compared to all the other boxes using the *Boxed Pareto dominance*. If dominated, this box is not refined;

d) Otherwise, the fidelity $l(\boldsymbol{x})$ is increased and the box is updated to $\mathcal{B}\big(\widetilde{\boldsymbol{f}}^l(\boldsymbol{x}), \widetilde{\boldsymbol{\varepsilon}}^l(\boldsymbol{x})\big)$ with the newly computed values $\widetilde{\boldsymbol{f}}^l(\boldsymbol{x})$ and $\widetilde{\boldsymbol{\varepsilon}}^l(\boldsymbol{x})$. This procedure is repeated as long as $\widetilde{\boldsymbol{\varepsilon}}^l(\boldsymbol{x}) \leq \boldsymbol{s}_2$, where $\boldsymbol{s}_2$ is a threshold fixed by the user. In the multi-objective case, $\boldsymbol{s}_2$ is a vector and the inequality holds for all dimensions;

e) After each refinement, the box is compared to the others using the *Boxed Pareto dominance*. If either the box gets dominated or $\widetilde{\boldsymbol{\varepsilon}}^l(\boldsymbol{x}) \leq \boldsymbol{s}_2$, it is no longer refined;

f) The value returned to the optimizer is the center of the box, *i.e.* $\widetilde{\boldsymbol{f}}^l(\boldsymbol{x})$.

Remarks

- When dealing with multiple new designs, for example with evolutionary algorithm, refinements (step d) are performed sequentially, one box at a time. This allows to check for domination at each refinement to avoid fidelity increases. This behavior is formally defined in Definition 10.
- Note that in principle, the inequality in step d) must not hold for all dimensions. Notably, when the objective functions are evaluated with different solvers, increasing the fidelity on one solver would not impact at all the fidelity of the objective functions assessed with another solver. As a consequence, each inequality should apply only to the set of objective functions evaluated with the same solver. As an underlying assumption, the scope of this paper is to handle one expensive solver, where the outcome of the simulation provides all the objective functions. For this reason, we consider the inequality to hold for all dimensions.
- Concerning the choice of $\boldsymbol{s}_2$, each component is written as $s_{2_k} = \bar{s}_{2_k} \times h_k$, where h_k is equal to:
$$h_k = \max_i \big(\widetilde{f}_k^l(\boldsymbol{x}_i)\big) - \min_i \big(\widetilde{f}_k^l(\boldsymbol{x}_i)\big),$$
given a set of estimations $\big\{\boldsymbol{x}_i, \widetilde{\boldsymbol{f}}^l(\boldsymbol{x}_i)\big\}_i$. The choice in terms of the threshold is then made on the normalized parameter $\bar{s}_{2_k}$, which represents a percentage of the range covered by the objectives on the evaluated designs, updated at each iteration. These ranges are initialized with the first estimations $\big\{\widetilde{\boldsymbol{f}}^0(\boldsymbol{x}_i)\big\}_i$ on an initial set of designs. In the following, the value of $\bar{\boldsymbol{s}}_2$ is given with a percent sign to recall this procedure.

Figure 2 illustrates an example of the output produced by the Bounding-Box approach in the case of a bi-objective minimization problem. We observe that green dominated designs (in the top right of the figure) feature large boxes, while red non-dominated designs are associated with smaller boxes.

This strategy is part of the flowchart (Fig. 3), and the blue color of the "Bounding-Box approach" box indicates the significant computational cost associated to the objectives estimation. This approach gives an efficient heuristic to allocate the estimation fidelity in order to i) ensure a high-fidelity computation of the Pareto front and ii) minimize the associated computational cost.

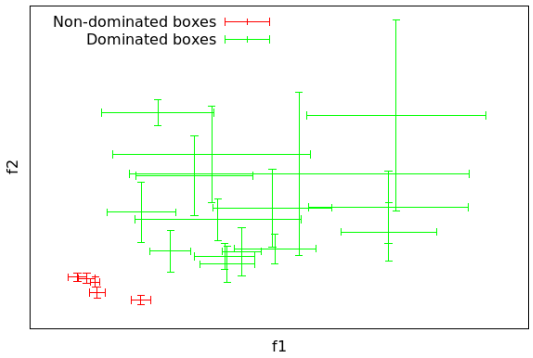


Figure 2: Bounding-Boxes refinement strategy for a bi-objective minimization problem

2.4 Surrogate-assisting strategy

In [22], the recursive Bounding-Box approach shows very good parsimony only during the first optimization iterations. Afterwards, most visited designs are located close to the Pareto front and require high-fidelity computations. In consequence, the strategy ends up behaving like a classical nested optimization.

For this reason, the Bounding-Box approach is coupled with a Surrogate-Assisting (SA) strategy. We propose here to build a surrogate model of the estimated objectives $\widetilde{\boldsymbol{f}}^l(\boldsymbol{x})$. This surrogate model is built on all previously estimated objectives $\left\{\widetilde{\boldsymbol{f}}^l(\boldsymbol{x}_i)\right\}_{i=1}^N$, where N is the number of boxes computed so far, and is used as a substitute for $\widetilde{\boldsymbol{f}}^l$ in areas where the surrogate error is low enough. The Surrogate-Assisting model is recomputed at each optimization iteration. We denote then the Surrogate-Assisting model at each iteration as $\boldsymbol{f}_{SA}^t$. During the last iterations, the surrogate model can be massively used, as the optimizer stabilizes in the optimal area. Practically, this approach should allow a faster densification of the approximated Pareto front.

The unknown surrogate error between $\boldsymbol{f}_{SA}^t(\boldsymbol{x})$ and $\boldsymbol{f}(\boldsymbol{x})$ at iteration t is denoted by $\boldsymbol{\varepsilon}_{SA}^t(\boldsymbol{x})$. It is approximated by a conservative approximation $\widetilde{\boldsymbol{\varepsilon}}_{SA}^t(\boldsymbol{x})$, so that $\widetilde{\boldsymbol{\varepsilon}}_{SA}^t(\boldsymbol{x}) \geq |\boldsymbol{\varepsilon}_{SA}^t(\boldsymbol{x})|$. We assume in this paper that we can estimate this error $\widetilde{\boldsymbol{\varepsilon}}_{SA}^t(\boldsymbol{x})$. For example, Bayesian regression approaches like Gaussian processes give a variability estimate of the surrogate model at each $\boldsymbol{x}$. Because of the estimation noise on the training points $\left(\boldsymbol{x}_i, \widetilde{\boldsymbol{f}}^l(\boldsymbol{x}_i)\right)$ on which the surrogate model is built, a regression model seems *a priori* more adequate than an interpolation.

When the optimizer provides a new design $\boldsymbol{x}$, the SA error $\widetilde{\boldsymbol{\varepsilon}}_{SA}^t(\boldsymbol{x})$ is compared to a user-defined threshold $\boldsymbol{s}_1$. If $\widetilde{\boldsymbol{\varepsilon}}_{SA}^t(\boldsymbol{x}) \leq \boldsymbol{s}_1$, the predictive value of the surrogate $\boldsymbol{f}_{SA}^t(\boldsymbol{x})$ is returned to

the optimizer. Else, the Bounding-Box approach is applied, and the associated box is computed and refined. Again, if $\boldsymbol{s}_1$ is multidimensional, inequality must hold for all dimensions. The SA model is updated with the knowledge of the newly computed box at each optimization iteration t. The value $\boldsymbol{f}_{opt}(\boldsymbol{x})$ returned to the optimizer depends then on the value of the surrogate error compared to a user-defined threshold $\boldsymbol{s}_1$. The choice of $\boldsymbol{s}_1$ follows the same procedure as for $\boldsymbol{s}_2$ (presented in Section 2.3). Namely, for each objective k, a normalized value $\bar{s}_{1_k} = \dfrac{s_{1_k}}{h_k}$ is provided by the user, with $h_k = \max_i\left(\widetilde{f}_k^l(\boldsymbol{x}_i)\right) - \min_i\left(\widetilde{f}_k^l(\boldsymbol{x}_i)\right)$. In summary,

$$\boldsymbol{f}_{opt}(\boldsymbol{x}) = \begin{cases} \boldsymbol{f}_{SA}^t(\boldsymbol{x}) & \text{if } \widetilde{\boldsymbol{\varepsilon}}_{SA}^t(\boldsymbol{x}) \leq \boldsymbol{s}_1 \ = \bar{\boldsymbol{s}}_1 \times \boldsymbol{h} \\ \widetilde{\boldsymbol{f}}^l(\boldsymbol{x}) & \text{else} \end{cases}. \tag{3}$$

The SA strategy and its coupling to the Bounding-Box approach are depicted in Figure 3, where the flowchart summarizes the behavior of the framework:

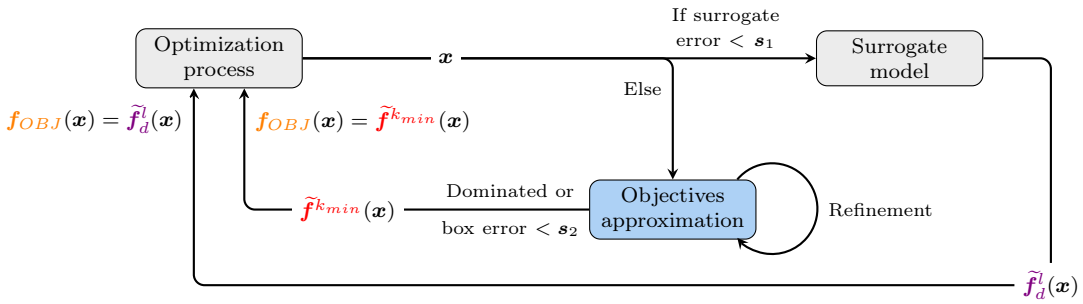


Figure 3: Structure of the SABBa framework

3 Mathematical analysis of the proposed approach

The goal of this section is to provide some proofs of convergence of the proposed strategy. In particular, first, we show how the designs produced with the Bounding-Box approach converge towards the exact Pareto-optimal area under some assumptions. Then, we illustrate the convergence of the method also when coupled with a Surrogate-Assisting strategy.

3.1 Bounding-Box definition and convergence analysis

Based on the Pareto dominance defined in the preceding section, the following Pareto-optimal sets are introduced. The set $\mathcal{P}$ represents the true continuous Pareto front of the problem. Its discrete counterpart is labeled as $\widetilde{\mathcal{P}}$. Finally, we indicate with $\widetilde{\mathcal{P}}_{\mathcal{B}}$ the set of Pareto-optimal boxes. Here, the tilda refers to the approximation arising from the discrete representation of the sets.

Definition 6. With $\mathcal{A} \subset \mathbb{R}^m$, designs $\boldsymbol{a}_i \in \mathbb{R}^m$ and boxes $\mathcal{B}_i \in \wp(\mathbb{R}^m)$:

$$\begin{aligned} \mathcal{P}(\mathcal{A}) &= \left\{\boldsymbol{a} \in \mathcal{A} \mid \nexists \boldsymbol{b} \in \mathcal{A},\ \boldsymbol{b} \succ \boldsymbol{a}\right\}, \\ \widetilde{\mathcal{P}}\left(\{\boldsymbol{a}_i\}_{i=1}^N\right) &= \left\{\boldsymbol{a}_i,\ i \in [\![1, N]\!] \mid \nexists j \in [\![1, N]\!],\ \boldsymbol{a}_j \succ \boldsymbol{a}_i\right\}, \\ \widetilde{\mathcal{P}}_{\mathcal{B}}\left(\{\mathcal{B}_i\}_{i=1}^N\right) &= \left\{\mathcal{B}_i,\ i \in [\![1, N]\!] \mid \nexists j \in [\![1, N]\!],\ \mathcal{B}_j \underset{\mathcal{B}}{\succ} \mathcal{B}_i\right\}. \end{aligned}$$

Ordinarily, $\mathcal{A}$ is simply $\boldsymbol{f}(\mathcal{X})$ and $\mathcal{P}(\boldsymbol{f}(\mathcal{X}))$ can simplify to $\mathcal{P}$, *i.e.* the Pareto front of the problem. On the contrary, the discrete $\widetilde{\mathcal{P}}$ always refers to a given set of designs.

Definition 7. With $\boldsymbol{f}$ the objective functions, $\mathcal{X}$ the design space, $\boldsymbol{x}_i \in \mathcal{X}$ and $\boldsymbol{r}_i \in \mathbb{R}^m$, let us define

$$\begin{aligned}
\mathcal{X}^{\boldsymbol{f}}_{\mathcal{P}} &= \big\{\boldsymbol{x} \in \mathcal{X} \mid \boldsymbol{f}(\boldsymbol{x}) \in \mathcal{P}\big\},\\
\mathcal{X}^{\boldsymbol{f}}_{\widetilde{\mathcal{P}}}\Big(\big\{\boldsymbol{x}_i\big\}_{i=1}^N\Big) &= \Big\{\boldsymbol{x}_i,\ i \in [\![1,N]\!] \mid \boldsymbol{f}(\boldsymbol{x}_i) \in \widetilde{\mathcal{P}}\Big(\big\{\boldsymbol{f}(\boldsymbol{x}_i)\big\}_{i=1}^N\Big)\Big\},\\
\mathcal{X}^{\boldsymbol{f}}_{\widetilde{\mathcal{P}}_{\mathcal{B}}}\Big(\big\{(\boldsymbol{x}_i,\boldsymbol{r}_i)\big\}_{i=1}^N\Big) &= \Big\{\boldsymbol{x}_i,\ i \in [\![1,N]\!] \mid \mathcal{B}_{\boldsymbol{f}}(\boldsymbol{x}_i,\boldsymbol{r}_i) \in \widetilde{\mathcal{P}}_{\mathcal{B}}\Big(\big\{\mathcal{B}_{\boldsymbol{f}}(\boldsymbol{x}_i,\boldsymbol{r}_i)\big\}_{i=1}^N\Big)\Big\}
\end{aligned}$$

Practically, $\mathcal{X}^{\boldsymbol{f}}_{\mathcal{P}}$, $\mathcal{X}^{\boldsymbol{f}}_{\widetilde{\mathcal{P}}}$ and $\mathcal{X}^{\boldsymbol{f}}_{\widetilde{\mathcal{P}}_{\mathcal{B}}}$ represent the ensemble of design coordinates associated with $\mathcal{P}$, $\widetilde{\mathcal{P}}$ and $\widetilde{\mathcal{P}}_{\mathcal{B}}$, respectively.

Remark $\mathcal{X}^{\boldsymbol{f}}_{\widetilde{\mathcal{P}}_{\mathcal{B}}}\Big(\big\{(\boldsymbol{x}_i,\boldsymbol{r}_i)\big\}_{i=1}^N\Big)$ can be explicitly defined as follows:

$$\boldsymbol{x}_i \in \mathcal{X}^{\boldsymbol{f}}_{\widetilde{\mathcal{P}}_{\mathcal{B}}}\Big(\big\{(\boldsymbol{x}_i,\boldsymbol{r}_i)\big\}_{i=1}^N\Big) \iff \forall k \in [\![1,N]\!], \exists j \in [\![1,m]\!],\ \pm f_j(\boldsymbol{x}_i) - r_{ij} < \pm f_j(\boldsymbol{x}_k) + r_{kj}. \quad (4)$$

Proposition 1. *Note that* $\forall \big\{(\boldsymbol{x}_i,\boldsymbol{r}_i)\big\}_{i=1}^N \in \big(\mathcal{X}\times\mathbb{R}^m\big)^N$,

$$\mathcal{X}^{\boldsymbol{f}}_{\widetilde{\mathcal{P}}}\Big(\big\{\boldsymbol{x}_i\big\}_{i=1}^N\Big) \subseteq \big\{\boldsymbol{x}_i\big\}_{i=1}^N \text{ and } \mathcal{X}^{\boldsymbol{f}}_{\widetilde{\mathcal{P}}_{\mathcal{B}}}\Big(\big\{(\boldsymbol{x}_i,\boldsymbol{r}_i)\big\}_{i=1}^N\Big) \subseteq \big\{\boldsymbol{x}_i\big\}_{i=1}^N.$$

The proof is provided in A.

Note that the inclusion becomes strict whenever a design does not satisfy the non-domination condition.

Two other sets are introduced, $\mathcal{P}_c$ and $\widetilde{\mathcal{P}}_c$, which represent the continuous extension of $\mathcal{P}$ and $\widetilde{\mathcal{P}}$, respectively. These sets contain all the dominated but not strictly dominated points of $\mathbb{R}^m$ with respect to $\mathcal{P}$ and $\widetilde{\mathcal{P}}$.

Definition 8. With $\mathcal{A} \subset \mathbb{R}^m$ and $\forall i, \boldsymbol{a}_i \in \mathbb{R}^m$, let us define $\mathcal{P}_c$ and $\widetilde{\mathcal{P}}_c$, as follows

$$\begin{aligned}
\mathcal{P}_c(\mathcal{A}) &= \mathcal{P}(\mathcal{A}) \cup \big\{\boldsymbol{b} \in \mathbb{R}^m \mid \exists \boldsymbol{z} \in \mathcal{P}(\mathcal{A}),\ \boldsymbol{z} \succ \boldsymbol{b}\ \&\ \nexists \boldsymbol{z} \in \mathcal{P}(\mathcal{A}),\ \boldsymbol{z} \succcurlyeq \boldsymbol{b}\big\},\\
\widetilde{\mathcal{P}}_c\Big(\big\{\boldsymbol{a}_i\big\}_{i=1}^N\Big) &= \widetilde{\mathcal{P}}\Big(\big\{\boldsymbol{a}_i\big\}_{i=1}^N\Big) \cup \Big\{\boldsymbol{b} \in \mathbb{R}^m \big| \exists \boldsymbol{z} \in \widetilde{\mathcal{P}}\Big(\big\{\boldsymbol{a}_i\big\}_{i=1}^N\Big), \boldsymbol{z} \overset{\mathcal{P}}{\succ} \boldsymbol{b}\ \&\ \nexists \boldsymbol{z} \in \widetilde{\mathcal{P}}\Big(\big\{\boldsymbol{a}_i\big\}_{i=1}^N\Big), \boldsymbol{z} \overset{\mathcal{P}}{\succcurlyeq} \boldsymbol{b}\Big\}.
\end{aligned}$$

Usually, as for Def. 6, $\mathcal{A} \equiv \boldsymbol{f}(\mathcal{X})$ and $\mathcal{P}_c(\mathcal{A})$ is denoted by $\mathcal{P}_c$.

3.1.1 Mathematical formulation for error-based boxes

As stated in Section 2, the proposed framework is developed in the context of estimated objective functions with tunable fidelity. The goal of this section is to provide some definitions and to introduce the so-called Bounding-Box recursive strategy.

Definition 9. An error is assumed on the computation of each objective function f_j. The estimated value $\widetilde{f}^l_j(\boldsymbol{x})$ with fidelity l differs from $f_j(\boldsymbol{x})$ by an additive error, as follows:

$$\forall \boldsymbol{x} \in \mathcal{X}, \forall j \in [\![1,m]\!],\ f_j(\boldsymbol{x}) = \widetilde{f}^l_j(\boldsymbol{x}) + \varepsilon^l_j(\boldsymbol{x}).$$

Remark Let us recall here that l implicitly refers to $l(\boldsymbol{x})$. The fidelity used for computing the objective functions can differ from one $\boldsymbol{x}$ to another. The aim of this work is to compute objective functions with high accuracy (high values of l) only for efficient designs.

Assumption 1. We assume that a conservative error $\widetilde{\boldsymbol{\varepsilon}}^l$ can be computed on $\widetilde{\boldsymbol{f}}^l$, meaning that:

$$\forall \boldsymbol{x} \in \mathcal{X}, \forall j \in [\![1,m]\!],\ |\varepsilon_j^l(\boldsymbol{x})| \leq \overline{\varepsilon}_j^l(\boldsymbol{x}).$$

Then, $\forall \boldsymbol{x} \in \mathcal{X}, \boldsymbol{f}(\boldsymbol{x}) \in \mathcal{B}_{\widetilde{\boldsymbol{f}}^l}\big(\boldsymbol{x}, |\boldsymbol{\varepsilon}^l(\boldsymbol{x})|\big) \subseteq \mathcal{B}_{\widetilde{\boldsymbol{f}}^l}\big(\boldsymbol{x}, \widetilde{\boldsymbol{\varepsilon}}^l(\boldsymbol{x})\big)$.

This assumption will be necessary for strictly demonstrating the convergence of the proposed approach.

The estimation $\widetilde{\boldsymbol{f}}^l$ can be refined by increasing the fidelity l. The conservative error is then assumed to converge to zero.

Assumption 2. No constraint is imposed on the monotony of $\big(\widetilde{\boldsymbol{\varepsilon}}^l\big)_{l\in\mathbb{N}_+}$ but it is assumed in the following that:

$$\forall j \in [\![1,m]\!], \lim_{l\to+\infty} \overline{\varepsilon}_j^l = 0.$$

Note that the computational cost increases with the fidelity.

In this work, the fidelity l is tuned adaptively for each design $\boldsymbol{x}$. Hence, we consider the fidelity l as a strictly increasing sequence at each $\boldsymbol{x}$ during the optimization process. At each refinement iteration, the fidelity of one estimation $\widetilde{\boldsymbol{f}}^l(\boldsymbol{x})$ is increased. The iteration is denoted with the subscript k. Hence, after k refinements, the fidelity at a given $\boldsymbol{x}$ is l_k and the estimated objectives value is $\widetilde{\boldsymbol{f}}^{l_k}(\boldsymbol{x})$.

The output of interest is $\mathcal{X}_{\mathcal{P}}^{\boldsymbol{f}}$. The proposed strategy should allow for an accurate discrete approximation of this set based on estimations $\widetilde{\boldsymbol{f}}^{l_k}$ with sequentially tuned fidelity l_k. To this extent, the classical and recursive approaches are presented. The later has been developed in [22]. Note that the following development is written in the design space, but can easily be transposed to the objective space.

Definition 10. For any $\{\boldsymbol{x}_i\}_{i=1}^N \in \mathcal{X}^N$, $\boldsymbol{f}$ the objective functions and $\widetilde{\boldsymbol{f}}^{l_k}$ the approximations with sequentially tuned fidelity l_k and conservative error $\widetilde{\boldsymbol{\varepsilon}}^{l_k}$, with $\big(N_k\big)_{k\in\mathbb{N}_+}$ a decreasing sequence and $N_0 = N$, let us introduce $\forall k \in \mathbb{N}_+$ the following sequences, initialized by $\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},0} = \{\boldsymbol{x}_i\}_{i=1}^N$.

$$\begin{aligned}
\text{Classical:}\quad \widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k+1} &= \mathcal{X}_{\widetilde{\mathcal{P}}_{\mathcal{B}}}^{\widetilde{\boldsymbol{f}}^{l_k}}\Big(\Big\{\Big(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}_i}^{\boldsymbol{f},0}, \widetilde{\boldsymbol{\varepsilon}}^{l_k}\Big(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}_i}^{\boldsymbol{f},0}\Big)\Big)\Big\}_{i=1}^{N}\Big),\\
\text{Recursive:}\quad \widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k+1} &= \mathcal{X}_{\widetilde{\mathcal{P}}_{\mathcal{B}}}^{\widetilde{\boldsymbol{f}}^{l_k}}\Big(\Big\{\Big(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}_i}^{\boldsymbol{f},k}, \widetilde{\boldsymbol{\varepsilon}}^{l_k}\Big(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}_i}^{\boldsymbol{f},k}\Big)\Big)\Big\}_{i=1}^{N_k}\Big).
\end{aligned} \tag{5}$$

$\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k}$ refers to the approximation of $\mathcal{X}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f}}$ at the k^{th} refinement iteration, using $\widetilde{\boldsymbol{f}}^{l_k}$ instead of $\boldsymbol{f}$. It represents the Pareto-optimal set associated to a given set of fidelities $\big\{l_k(\boldsymbol{x}_i)\big\}_i$.

Remark In the following, by default, $\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k}$ will refer to the recursive strategy.

The classical approach refers to a double-loop or nested optimization, where the objective functions are estimated with increasing fidelity on the whole design set $\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},0}$ with size N. As raised in Assumption 2, the most time-consuming step is the decrease of the conservative error $\widetilde{\boldsymbol{\varepsilon}}^l$ through higher fidelity computations. In the recursive Bounding-Box refinement strategy, Proposition 1 gives that $\forall k \in \mathbb{N}_+,\ \widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k} \subseteq \widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k-1} \subseteq ... \subseteq \widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},0}$. Any unsatisfied non-domination

condition implies a strict inclusion between two sets of the sequence. Hence, whenever a box gets dominated, the size N_k $\left(= Card\big(\widetilde{\mathcal{X}}^{\boldsymbol{f},k}_{\widetilde{\mathcal{P}}}\big)\right)$ of the set of interest is decreased. The fidelity increased is then performed only on $N_k \leq N$ designs.

The next section is devoted to the convergence of the recursive set sequence.

3.1.2 Convergence analysis

This section aims to demonstrate that the sequence $\big(\widetilde{\mathcal{X}}^{\boldsymbol{f},k}_{\widetilde{\mathcal{P}}}\big)_{k\in\mathbb{N}_+}$ theoretically converges to the real Pareto-optimal set. More precisely, the goal is to demonstrate the convergence of the continuous approximated Pareto front $\widetilde{\mathcal{P}}_c$ built on the approximated objectives $\widetilde{\boldsymbol{f}}^{l_k}\big(\widetilde{\mathcal{X}}^{\boldsymbol{f},k}_{\widetilde{\mathcal{P}}}\big)$ toward the real continuous front $\mathcal{P}_c$.

To this extent, another assumption is needed.

Assumption 3. For a given multi-objective optimization, it is assumed that:

$$\forall \boldsymbol{y} \in \mathcal{X}^{\boldsymbol{f}}_{\mathcal{P}}, \exists \mathcal{D} \subseteq \mathcal{X} \text{ so that } \boldsymbol{y} \in \mathcal{D}, \mathring{\mathcal{D}} \neq \emptyset \text{ and } \forall j \in [\![1,m]\!],\ f_j \in \mathcal{C}^0(\mathcal{D}).$$

Let $\mathcal{D}(\boldsymbol{y})$ be a set satisfying the previous condition for a given $\boldsymbol{y} \in \mathcal{X}$.

$$\forall \epsilon \in \mathbb{R}^*_+, \exists M \in \mathbb{N}_+, \forall \boldsymbol{y} \in \mathcal{X}^{\boldsymbol{f}}_{\mathcal{P}}, \exists k \in [\![1,m]\!],\ \boldsymbol{x}_k \in \mathcal{D}(\boldsymbol{y}), \big\|\boldsymbol{x}_k - \boldsymbol{y}\big\| \leq \epsilon.$$

In practice, it states that the optimizer converges toward the whole Pareto front and discretely covers the integrity of $\mathcal{X}^{\boldsymbol{f}}_{\mathcal{P}}$. In addition, there exists a non-empty part of $\mathcal{X}$ around each efficient design in which all f_j are continuous.

These assumptions simulate a "well-behaving" optimizer. In this context, the impact of the proposed strategy on the convergence can be studied.

Remarks The following relation is trivial:

$$\mathcal{X}^{\boldsymbol{f}}_{\widetilde{\mathcal{P}}}\Big(\mathcal{X}^{\boldsymbol{f}}_{\widetilde{\mathcal{P}}}\Big(\big\{\boldsymbol{x}_i\big\}_{i=1}^N\Big)\Big) = \mathcal{X}^{\boldsymbol{f}}_{\widetilde{\mathcal{P}}}\Big(\big\{\boldsymbol{x}_i\big\}_{i=1}^N\Big).$$

More generally, $\forall \mathcal{S}$ s.t. $\mathcal{X}^{\boldsymbol{f}}_{\widetilde{\mathcal{P}}}\Big(\big\{\boldsymbol{x}_i\big\}_{i=1}^N\Big) \subseteq \mathcal{S} \subseteq \big\{\boldsymbol{x}_i\big\}_{i=1}^N$,

$$\mathcal{X}^{\boldsymbol{f}}_{\widetilde{\mathcal{P}}}(\mathcal{S}) = \mathcal{X}^{\boldsymbol{f}}_{\widetilde{\mathcal{P}}}\Big(\big\{\boldsymbol{x}_i\big\}_{i=1}^N\Big). \tag{6}$$

This can be transposed for $\widetilde{\mathcal{P}}$ in the objective space.

Proposition 2. $\forall \boldsymbol{y} \in \big\{\boldsymbol{x}_i\big\}_{i=1}^N,\ \ \boldsymbol{y} \notin \mathcal{X}^{\boldsymbol{f}}_{\widetilde{\mathcal{P}}}\Big(\big\{\boldsymbol{x}_i\big\}_{i=1}^N\Big) \iff \exists \boldsymbol{y}' \in \mathcal{X}^{\boldsymbol{f}}_{\widetilde{\mathcal{P}}}\Big(\big\{\boldsymbol{x}_i\big\}_{i=1}^N\Big),\ \boldsymbol{y}' \succ \boldsymbol{y}$.

The proof is given in B.

Lemma 1. $\forall \big\{\big(\boldsymbol{x}_i, \widetilde{\boldsymbol{\varepsilon}}(\boldsymbol{x}_i)\big)\big\}_{i=1}^N \in \big(\mathcal{X}\times\mathbb{R}^m\big)^N,\ \mathcal{X}^{\boldsymbol{f}}_{\widetilde{\mathcal{P}}}\Big(\big\{\boldsymbol{x}_i\big\}_{i=1}^N\Big) \subseteq \mathcal{X}^{\boldsymbol{f}}_{\widetilde{\mathcal{P}}_{\mathcal{B}}}\Big(\big\{\big(\boldsymbol{x}_i, \widetilde{\boldsymbol{\varepsilon}}(\boldsymbol{x}_i)\big)\big\}_{i=1}^N\Big)$.

The proof is given in C.

Proposition 2 simply formalizes that there is at least one dominant design in a given set of points. Lemma 1 shows the robustness of the boxed dominance. All truly efficient design are also efficient in the *Boxed Pareto dominance* sense.

A novelty of this paper compared to Fusi's Bounding-Box approach in [22] is the choice of an accuracy threshold for the estimation of the objective functions. In [22], *exact* (or very refined) estimations are computed for non-dominated designs. Here, non-dominated boxes are only refined up to a given user-defined threshold.

Theorem 1. *For any initial set* $\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},0} = \{\boldsymbol{x}_i\}_{i=1}^{N} \in \mathcal{X}^N$*, approximation sequence* $\left(\widetilde{\boldsymbol{f}}^{l_k}\right)_{k\in\mathbb{N}_+}$ *and conservative error set sequence* $\left(\widetilde{\boldsymbol{\varepsilon}}^{l_k}\right)_{k\in\mathbb{N}_+}$*, the robustness is verified* $\forall k \in \mathbb{N}_+$*, i.e.:*

$$\mathcal{X}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f}}\Big(\{\boldsymbol{x}_i\}_{i=1}^{N}\Big) \subseteq \widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k}. \tag{7}$$

The proof is given in D.

This first theorem extends the robust behavior presented in Lemma 1 to the recursive Bounding-Box strategy introduced in Definition 10. We raised that this strategy allows to reduce the number of refinements, Theorem 1 also ensures that all Pareto-optimal designs are retained for high-fidelity estimation.

Using Assumption 3, the convergence of $\widetilde{\mathcal{P}}_c\big(\widetilde{\boldsymbol{f}}^{l_k}\big(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k}\big)\big)$ toward the real continuous front $\mathcal{P}_c$ can be proven, as follows.

Theorem 2. *For any initial set* $\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},0} \in \mathcal{X}^N$*, approximation sequence* $\left(\widetilde{\boldsymbol{f}}^{l_k}\right)_{k\in\mathbb{N}_+}$ *and conservative error set sequence* $\left(\widetilde{\boldsymbol{\varepsilon}}^{l_k}\right)_{k\in\mathbb{N}_+}$*, the following convergence is verified:*

$$\lim_{(N,k)\to(+\infty,+\infty)} d_H\Big(\widetilde{\mathcal{P}}_c\big(\widetilde{\boldsymbol{f}}^{l_k}\big(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k}\big)\big),\ \mathcal{P}_c\Big) = 0, \tag{8}$$

with the classical Hausdorff distance denoted by d_H*.*

The proof is given in E.

The continuity assumption brings preimage convergence.

In this section, the convergence analysis has been carried out. It relies on several assumptions on the computability of conservative errors and the good behavior of both the optimizer and the objective functions. The recursive set sequence is the set of non-dominated designs, which cardinality decreases when boxes get dominated. Performing refinements only on this set brings significant cost reduction, notably during the exploration phase of the optimizer.

Contrarily to [22], the accuracy of the efficient boxes is tuned with a user-defined threshold $\boldsymbol{s}_2$. The fidelity of the objectives estimation is adaptively increased until $\widetilde{\boldsymbol{\varepsilon}}^{l}(\boldsymbol{x}_i) \leq \boldsymbol{s}_2$ for all $\boldsymbol{x}_i$. In the multi-objective case, $\boldsymbol{s}_2$ is a user-defined vector and the notation $\boldsymbol{a} \leq \boldsymbol{b}$ means $\forall i, a_i \leq b_i$. This approach has been proven to be conservative, meaning that truly efficient designs are not discarded during the recursive set sequence refinement.

3.2 Surrogate-Assisting model

In this section, we study the coupling between the Bounding-Box approach and a Surrogate-Assisting (SA) strategy. The later should not impact the converge towards the true Pareto optima. We recall the notation introduced in Section 2.4.

The SA model predictive value at iteration t is denoted $\boldsymbol{f}_{SA}^{t}(\boldsymbol{x})$ and the known conservative error is $\widetilde{\boldsymbol{\varepsilon}}_{SA}^{t}(\boldsymbol{x}) \geq \boldsymbol{\varepsilon}_{SA}^{t}(\boldsymbol{x}) = \boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}_{SA}^{t}(\boldsymbol{x})$. The value returned to the optimizer is $f_{opt}(\boldsymbol{x})$, that uses the SA model predictive value when accuracy $\boldsymbol{s}_1$ is reached, and the converged Bounding-Box estimation otherwise, see Equation (3).

Remark Again, in the case of multiple objectives $m > 1$, the surrogate errors and the user-defined vector $\boldsymbol{s}_1$ must be compared. They are both of size m and:

$$\boldsymbol{a} \leq \boldsymbol{b} \iff \forall i, a_i \leq b_i.$$

Theorem 3. *Given two user-defined thresholds $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$, the true discrete Pareto optima are included in the following boxed Pareto optima:*

$$\forall \{\boldsymbol{x}_i\}_{i=1}^N \in \mathcal{X}^N,\ \mathcal{X}^{\boldsymbol{f}}_{\widetilde{\mathcal{P}}}\Big(\{\boldsymbol{x}_i\}_{i=1}^N\Big) \subseteq \mathcal{X}^{\boldsymbol{f}_{opt}}_{\widetilde{\mathcal{P}}_{\mathcal{B}}}\Big(\big\{\big(\boldsymbol{x}_i, \boldsymbol{r}(\boldsymbol{x}_i)\big)\big\}_{i=1}^N\Big),$$

where equality holds when $\boldsymbol{s}_1 = \boldsymbol{s}_2 = \boldsymbol{0}$. The half-width $\boldsymbol{r}$ depends on the use or not of the Surrogate-Assisting model:

$$\boldsymbol{r}(\boldsymbol{x}_i) = \begin{cases} \widetilde{\boldsymbol{\varepsilon}}^t_{SA}(\boldsymbol{x}_i) & \text{if } \widetilde{\boldsymbol{\varepsilon}}^t_{SA}(\boldsymbol{x}_i) \leq \boldsymbol{s}_1 \\ \widetilde{\boldsymbol{\varepsilon}}^l(\boldsymbol{x}_i) & \text{else} \end{cases}. \tag{9}$$

The proof is given in F.

Note that the surrogate model is iteratively refined throughout the optimization process. Hence, sequential optimization approaches will benefit the most from the Surrogate-Assisting strategy.

A sketch about the coupling is provided in Algorithm 1. Steps *(a)* to *(c)* will be detailed in Section 4.

Algorithm 1 Algorithm overview

```
while Optimization running do
    Read new designs
    if SA error below s_1 then
        (a) Use SA model to approximate objective functions
    else
        (b) Compute and (c) refine the recursive set sequence up to s_2
        Update the SA model
    end if
end while
```

The Surrogate-Assisting strategy complements efficiently the Bounding-Box approach. If a pattern, linearity or simple behavior of the objective functions is detected, it can drastically reduce the overall optimization cost. The convergence of the coupled approach can be deduced from Theorems 2 and 3, by replacing $N \to +\infty$ with $t \to +\infty$ and $k \to +\infty$ with $(\boldsymbol{s}_1, \boldsymbol{s}_2) \to (\boldsymbol{0}, \boldsymbol{0})$.

In the following, a numerical algorithm is proposed and applied to some analytical test-cases in order to validate the framework.

4 Algorithm of the framework

This section illustrates the numerical algorithm associated with the proposed framework.

As seen in the previous sections, two user-defined thresholds $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$ have to be chosen. The value $\boldsymbol{s}_1$ is compared to the surrogate error to determine whether the surrogate can be used or not. The threshold $\boldsymbol{s}_2$ is compared to the error of the estimated objective functions, to decide whether higher higher-fidelity estimations are required.

Three design sets $\boldsymbol{\mathcal{D}}$, $\boldsymbol{\mathcal{D}}_{new}$ and $\boldsymbol{\mathcal{D}}'$ are introduced for Algorithm 2. The set $\boldsymbol{\mathcal{D}}$ is initialized at the beginning of the algorithm. It contains all the non-dominated designs and is updated at each refinement of the objective functions. It corresponds to the current set of the recursive set sequence from Definition 10. The set $\boldsymbol{\mathcal{D}}_{new}$ represents the new designs at each optimization

iteration. It can contain several designs (*e.g.* with a genetic algorithm optimizer) or only one design (for sequential optimization). The set $\boldsymbol{\mathcal{D}'}$ contains the non-dominated designs of $\boldsymbol{\mathcal{D}}_{new}$ that were not computed with the SA strategy. This set is updated at each refinement and is reinitialized at each optimization iteration.

The main steps of Algorithm 2 are the following: i) in the subroutine `compareDesignSA`, if the surrogate error is below $\boldsymbol{s}_1$, the associated objective values are returned. A low-fidelity objective estimation is computed for the other designs and added to $\boldsymbol{\mathcal{D}'}$; ii) the inner while loop refines each design of $\boldsymbol{\mathcal{D}'}$ up to the $\boldsymbol{s}_2$ threshold within the `iterateRefinement` subroutine and checks for new dominations at each iteration to discard boxes from $\boldsymbol{\mathcal{D}'}$; iii) finally, the surrogate is updated and new designs can be analyzed.

Algorithm 2 Detailed pseudo-algorithm of the framework

Set $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$
Initialize $\boldsymbol{\mathcal{D}}$ empty
$t = 0$
while Optimization running **do**
  Read new designs $\boldsymbol{\mathcal{D}}_{new} = \left\{\boldsymbol{x}_i\right\}_{i=1}^{N}$
  $\boldsymbol{\mathcal{D}} = \boldsymbol{\mathcal{D}} \cup \boldsymbol{\mathcal{D}}_{new}$
  Initialize $\boldsymbol{\mathcal{D}'}$ empty
  $\boldsymbol{\mathcal{D}'}, \widetilde{\boldsymbol{f}}^0(\boldsymbol{\mathcal{D}'}), \widetilde{\boldsymbol{\varepsilon}}^0(\boldsymbol{\mathcal{D}'}), \boldsymbol{f}_{opt}, \boldsymbol{r} = \texttt{compareDesignSA}(\boldsymbol{\mathcal{D}}_{new}, \boldsymbol{s}_1, \boldsymbol{s}_2, \boldsymbol{f}_{SA}^t, \widetilde{\boldsymbol{\varepsilon}}_{SA}^t)$ (Alg. 3) ▷ *(a,b)*
  Compute $\boldsymbol{\mathcal{D}} = \widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f}_{opt},0} = \mathcal{X}_{\widetilde{\mathcal{P}}_{\mathcal{B}}}^{\boldsymbol{f}_{opt}}\big((\boldsymbol{\mathcal{D}}, \boldsymbol{r}(\boldsymbol{\mathcal{D}}))\big)$
  $\boldsymbol{\mathcal{D}'} = \boldsymbol{\mathcal{D}} \cap \boldsymbol{\mathcal{D}'}$
  $k = 0$
  while $\max\left(\widetilde{\boldsymbol{\varepsilon}}^l(\boldsymbol{x})\right) > \boldsymbol{s}_2,\ \boldsymbol{x} \in \boldsymbol{\mathcal{D}'}$ **do**
    $k = k + 1$
    Select $\boldsymbol{x}_r$ to refine in $\boldsymbol{\mathcal{D}'}$
    $\widetilde{\boldsymbol{f}}^{l_k}, \widetilde{\boldsymbol{\varepsilon}}^{l_k} = \texttt{iterateRefinement}(\boldsymbol{\mathcal{D}'}, \boldsymbol{x}_r)$ (Alg. 4) ▷ *(c)*
    for each $\boldsymbol{x} \in \boldsymbol{\mathcal{D}'}$ **do**
      $\boldsymbol{f}_{opt}(\boldsymbol{x}) = \widetilde{\boldsymbol{f}}^{l_k}(\boldsymbol{x})$
      $\boldsymbol{r}(\boldsymbol{x}) = \widetilde{\boldsymbol{\varepsilon}}^{l_k}(\boldsymbol{x})$
    end for
    Compute $\boldsymbol{\mathcal{D}} = \widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f}_{opt},k} = \mathcal{X}_{\widetilde{\mathcal{P}}_{\mathcal{B}}}^{\boldsymbol{f}_{opt}}\Big(\Big(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f}_{opt},k-1}, \boldsymbol{r}\big(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f}_{opt},k-1}\big)\Big)\Big)$
    $\boldsymbol{\mathcal{D}'} = \boldsymbol{\mathcal{D}} \cap \boldsymbol{\mathcal{D}'}$
  end while
  Compute updated $\boldsymbol{f}_{SA}^t$ and $\widetilde{\boldsymbol{\varepsilon}}_{SA}^t$
end while
Return $\boldsymbol{\mathcal{D}}$ and $\boldsymbol{\mathcal{B}}_{\boldsymbol{f}_{opt}}\big(\boldsymbol{\mathcal{D}}, \boldsymbol{r}(\boldsymbol{\mathcal{D}})\big)$

Remark In Algorithm 2, $\left\{\mathcal{B}_{\boldsymbol{f}_{opt}}\big(\mathcal{D}_i, \boldsymbol{r}(\mathcal{D}_i)\big)\right\}_{i=1}^{Card(\boldsymbol{\mathcal{D}})}$ is simply written $\boldsymbol{\mathcal{B}}_{\boldsymbol{f}_{opt}}\big(\boldsymbol{\mathcal{D}}, \boldsymbol{r}(\boldsymbol{\mathcal{D}})\big)$.

The subroutine `compareDesignSA` (in Algorithm 3) takes as inputs the new designs, the thresholds and the current SA model. It returns the set of non-dominated designs that can be refined (not coming from SA approximation), the associated low-fidelity estimation of the objective functions and the coupled objective functions $\boldsymbol{f}_{opt}$ with boxes size $\boldsymbol{r}$.

For each new design, the error of the SA model is compared to the threshold $\boldsymbol{s}_1$. If the error is small enough *(a)*, the objective functions are approximated with the surrogate. Otherwise *(b)*, low-fidelity estimations are computed and the design is added to $\boldsymbol{\mathcal{D}'}$. In practice, computation

(b) is assumed to have a much higher cost than computation *(a)*.

Algorithm 3 `compareDesignSA`$(\boldsymbol{\mathcal{D}}_{new}, \boldsymbol{s}_1, \boldsymbol{s}_2, \boldsymbol{f}_{SA}, \widetilde{\boldsymbol{\varepsilon}}_{SA})$

for each $\boldsymbol{x} \in \boldsymbol{\mathcal{D}}_{new}$ **do**
  if $\widetilde{\boldsymbol{\varepsilon}}_{SA}(\boldsymbol{x}) \leq \boldsymbol{s}_1$ **then** ▷ *(a)*
    $\boldsymbol{f}_{opt}(\boldsymbol{x}) = \boldsymbol{f}_{SA}(\boldsymbol{x})$
    $\boldsymbol{r}(\boldsymbol{x}) = \widetilde{\boldsymbol{s}_2}(\boldsymbol{x}) + \widetilde{\boldsymbol{\varepsilon}}_{SA}(\boldsymbol{x})$
  else ▷ *(b)*
    Compute $\widetilde{\boldsymbol{f}}^0(\boldsymbol{x})$ and $\widetilde{\boldsymbol{\varepsilon}}^0(\boldsymbol{x})$
    $\boldsymbol{f}_{opt}(\boldsymbol{x}) = \widetilde{\boldsymbol{f}}^0(\boldsymbol{x})$
    $\boldsymbol{r}(\boldsymbol{x}) = \widetilde{\boldsymbol{\varepsilon}}^0(\boldsymbol{x})$
    $\boldsymbol{\mathcal{D}}' = \boldsymbol{\mathcal{D}}' \cup \boldsymbol{x}$
  end if
end for
Return $\boldsymbol{\mathcal{D}}'$, $\widetilde{\boldsymbol{f}}^0(\boldsymbol{\mathcal{D}}')$, $\widetilde{\boldsymbol{\varepsilon}}^0(\boldsymbol{\mathcal{D}}')$, $\boldsymbol{f}_{opt}$ and $\boldsymbol{r}$

Then, `iterateRefinement` (in Algorithm 4) computes the new approximation $\widetilde{\boldsymbol{f}}^{l_k}$ and its associated error for the chosen design $\boldsymbol{x}_r$. The fidelity l_k associated to the design to refine $\boldsymbol{x}_r$ is increased and the new estimation is computed. For all other designs, the fidelity is kept constant. We chose to refine the boxes one at a time to check for domination at each iteration.

Algorithm 4 `iterateRefinement`$(\boldsymbol{\mathcal{D}}', \boldsymbol{x}_r)$

for each $\boldsymbol{x} \in \boldsymbol{\mathcal{D}}'$ **do**
  if $\boldsymbol{x} = \boldsymbol{x}_r$ **then**
    $l_k(\boldsymbol{x}) = l_{k-1}(\boldsymbol{x}) + 1$
    Compute higher-fidelity estimations $\widetilde{\boldsymbol{f}}^{l_k}(\boldsymbol{x})$ and $\widetilde{\boldsymbol{\varepsilon}}^{l_k}(\boldsymbol{x})$ ▷ *(c)*
  else
    $l_k(\boldsymbol{x}) = l_{k-1}(\boldsymbol{x})$
  end if
end for
Return $\widetilde{\boldsymbol{f}}^{l_k}$ and $\widetilde{\boldsymbol{\varepsilon}}^{l_k}$

5 Test-cases and analysis

The proposed framework (SABBa) is an adaptive strategy for approximating the Pareto front of an optimization problem by tuning the fidelity of the objectives estimations according to their efficiency to significantly lower the computational burden.

In this section, we apply the framework to several analytical test-cases and analyse its performance. The aim here is to provide a proof-of-concept of the proposed framework on analytical test-cases, *i.e.* to study the asymptotic behavior of SABBa compared to Fusi's approach [22] and the simple nested approach, that can be surrogate-assisted. For this reason, we do not choose any specific surrogate model, optimizer or multi-fidelity objective estimations, but we simulate them analytically. Note also that in this perspective, the point is not to assess the behavior of the framework concerning the dimensionality and complexity of the optimization problem, as these last ones impact the optimizer and the surrogate model, which are choices independent of the framework.

Section 5.1 illustrates how the multi-fidelity estimations, surrogate models, and the optimizer are estimated. The first test-case, treated in Section 5.2, is a classical bi-objective optimization. The second test-case illustrated in Section 5.3 highlights the increase of the computational cost raised in [22] when the optimizer converges toward the Pareto front. It permits to emphasize the gains induced by the Surrogate-Assisting (SA) strategy. Finally, the robustness of SABBa to complex Pareto front is illustrated in a third test-case, presented in Section 5.4.

5.1 Tools

As stated above, the different tools (optimizer, surrogate model and multi-fidelity computations) are simulated analytically. For real applications, any existing method can be chosen.

Multi-fidelity computations In this section, we deal with analytical test-cases. We add arbitrarily a noise on the real value, inversely proportional to the desired fidelity. In practice, the low-fidelity estimation costs one evaluation. Each refinement then increases this value by one. The first low-fidelity computation is performed as follows for each objective function f_i:

$$\begin{aligned}
\varepsilon_{noise_i} &= \frac{\max(f_i) - \min(f_i)}{4 + 6X_1} \quad \text{with } X_1 \sim \mathcal{U}[0,1], \\
\varepsilon_{bias_i} &= \varepsilon_{noise_i} X_2 \quad \text{with } X_2 \sim \mathcal{U}[-1,1], \\
\widetilde{f}_i^0 &= f_i + \varepsilon_{bias_i}, \\
\overline{\varepsilon}_i^0 &= 2.\varepsilon_{noise_i}.
\end{aligned}$$

This estimation can then be refined with a given refinement factor $r_f > 1$:

$$\begin{aligned}
\widetilde{f}_i^l &= f_i + \frac{\widetilde{f}_i^{l-1}}{-} f_i r_f, \\
\overline{\varepsilon}_i^l &= \frac{\overline{\varepsilon}_i^{l-1}}{r_f}.
\end{aligned}$$

Optimizer The optimization is performed with a sequence of Monte-Carlo samplings. At each iteration, the set of new designs is constituted by ten random designs. Convergence toward the optimal area can be enforced on the Monte Carlo samples. The effect of this convergence is studied in test-cases 2 and 3.

Surrogate model Similarly to the multi-fidelity computations, the Surrogate-Assisting (SA) model $\boldsymbol{f}_{SA}$ is an arbitrarily noisy evaluation of the objective functions. We consider a noise proportional to the distance $d_{\min}$ to the closest training point. The noise is also proportional to the range covered by the objective functions. Practically, at a given optimization iteration t and for each dimension i:

$$\begin{aligned}
\overline{\epsilon}_{SA_i}^t &= \frac{d_{\min}}{\frac{1}{N_x}\sum_j \big(\max(x_j) - \min(x_j)\big)} . \frac{\max(f_i) - \min(f_i)}{2}, \\
\varepsilon_{noise_i} &= \frac{\overline{\epsilon}_{SA_i}^t}{2} X_1 \quad \text{with } X_1 \sim \mathcal{U}[0,1], \\
\varepsilon_{bias_i} &= \varepsilon_{noise_i} X_2 \quad \text{with } X_2 \sim \mathcal{U}[-1,1], \\
f_{SA_i}^t &= f_i + \varepsilon_{bias_i}.
\end{aligned}$$

To simulate the fact that the surrogate is built on noisy training data, and to be consistent at least on the Pareto front, we add a constant noise on the conservative errors, as follows:

$$\bar{\varepsilon}^t_{SA_i} = \bar{\epsilon}^t_{SA_i} + \boldsymbol{s}_2.$$

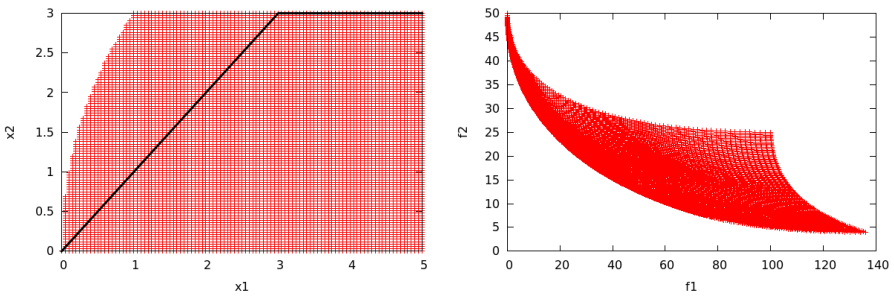
In the following, the SABBa framework is compared to three different approaches: i) the nested classical approach (denoted by *Class*), *i.e.* non-recursive Bounding-Box approach defined in Def. 10, where each box is refined up to the threshold $\boldsymbol{s}_2$; ii) the recursive approach, *i.e.* Fusi's Bounding-Box refinement strategy [22], where only non-dominated boxes are refined to the threshold $\boldsymbol{s}_2$; iii) a surrogate-assisted classical approach (denoted by *Class-acc*).

5.2 Test-case 1: The BNH problem

The first test-case deals with a bi-objective constrained problem, proposed by Binh and Korn (1997):

$$\begin{aligned} \text{minimize:} \quad & \boldsymbol{f}(\boldsymbol{x}) = \begin{pmatrix} 4x_1^2 + 4x_2^2 \\ (x_1-5)^2 + (x_2-5)^2 \end{pmatrix} \\ \text{subject to:} \quad & (x_1-5)^2 + x_2^2 \leq 25 \\ & (x_1-8)^2 + (x_2+3)^2 \geq 7.7 \\ \text{by changing:} \quad & (x_1, x_2) \in [0,5] \times [0,3] \end{aligned} \tag{10}$$

The acceptable design area and its counterpart in the objective space are represented in Fig. 4. The Pareto optimal set in the design space is represented by the thick black curve. New designs are chosen randomly in $[0,5] \times [0,3]$ under the optimization geometric constraints.

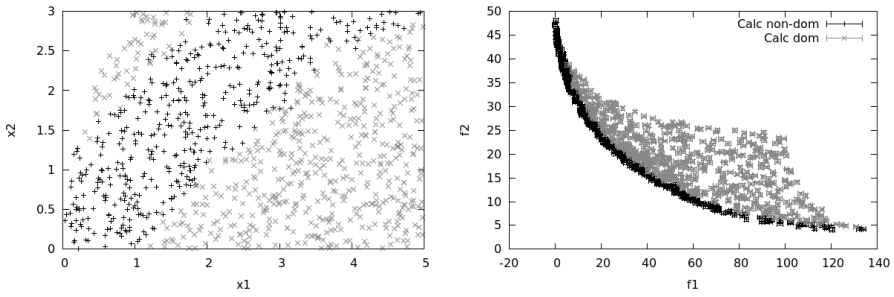


(a) (b)

Figure 4: Test-case 1, a) acceptable design and Pareto optima (thick line), b) image in the objective space.

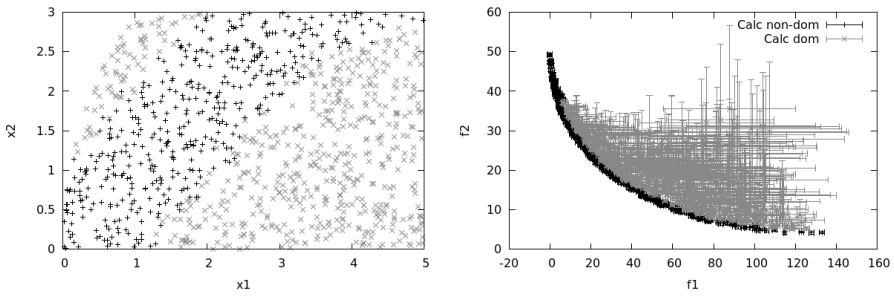
In Figures 5 and 6, the results in terms of optimal designs and objectives, for the classical and the recursive approaches are provided, respectively. This optimization is performed with 100 optimization iterations and $\bar{\boldsymbol{s}}_2 = (1\%, 1\%)^T$. This is simply written $\bar{\boldsymbol{s}}_2 = 1\%$ as we chose to use the same threshold for both objectives. We recall that the normalized thresholds $\bar{\boldsymbol{s}}_1$ and $\bar{\boldsymbol{s}}_2$ are multiplied by the range covered by each objective to obtain the thresholds $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$, as defined in Section 2.3 and 2.4. Because of the low gradient around the optima, the non-dominated designs (in black) cover a quite wide area. Dominated designs and boxes are drawn in grey.

The approximated Pareto front of these two strategies are qualitatively very similar and both provided the same Pareto-optimal design area. However, a significant portion of the boxes are



(a) (b)

Figure 5: Test-case 1, *Class approach*, non-dominated designs in black and dominated ones in grey: a) design space, b) objective space.



(a) (b)

Figure 6: Test-case 1, *Fusi approach*, non-dominated designs in black and dominated one in grey: a) design space, b) objective space.

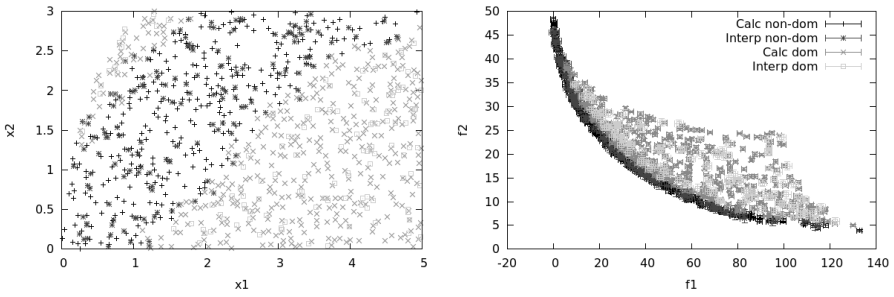
not refined up to $\boldsymbol{s}_2$ in the recursive approach (Figure 6), thus reducing the number of function evaluations, and the computational cost of the optimization.

In Figures 7 and 8, the Surrogate-Assisting (SA) strategy is applied on the classical approach and the recursive (Fusi) approach, respectively. Figure 8 pictures the behavior of SABBa. The thresholds $\bar{\boldsymbol{s}}_2 = 1\%$ and $\bar{\boldsymbol{s}}_1 = 1\%$ are used (implicitly $(1\%, 1\%)^T$).

SA designs and boxes are drawn in dark grey when they are non-dominated and light grey otherwise. The non-dominated design area and the approximated Pareto front are similar to the fully-computed ones from Figure 5.

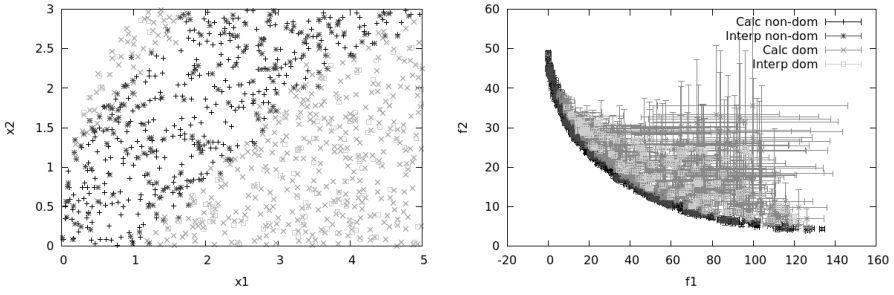
We directly compare the number of function evaluations needed for each strategy. Because the tools are analytically simulated, the number of function evaluations only allows for asymptotic comparison and should not be confronted to real applications. The results are reported in Figure 9a:

- The recursive (Fusi [22]) strategy appears to be more parsimonious than the classical approach. It reduces the slope of the cost curve by a ratio that can be interpreted as the mean computational cost in the recursive strategy with regards to the full convergence cost. Nearly half of the designs are non-dominated in this test-case. This high number of fully refined boxes implies limited computational gains.



(a) (b)

Figure 7: Test-case 1, *Class-acc approach*, non-dominated interpolated designs added in dark grey, dominated interpolated designs in light grey. a) design space, b) objective space.



(a) (b)

Figure 8: Test-case 1, *SABBa approach*, non-dominated interpolated designs added in dark grey, dominated interpolated designs in light grey: a) design space, b) objective space.

- The SA strategy is more and more effective throughout the optimization process. During the first iterations, the surrogate is not yet accurate and the slope is similar to non-assisted strategies. Later, surrogate-assisted strategies show significant cost reduction.

The influence of the user-defined thresholds $\bar{\boldsymbol{s}}_1$ and $\bar{\boldsymbol{s}}_2$ is also investigated.

Figure 9b pictures the substantial impact of $\bar{\boldsymbol{s}}_2$ on the computational cost. This comes from the high number of high-fidelity boxes. Increasing the threshold implies higher accuracy for all these boxes and significantly rises the global cost. However, this threshold has a critical repercussion on the accuracy of the Pareto-optimal set. This is qualitatively drawn in Fig. 10.

Figures 9c and 9d illustrate the momentous cost reduction that can be achieved with the SA strategy, at the expense of Pareto front accuracy. One should however be careful not to underestimate the surrogate error, *e.g.* when high local variations are not yet captured.

The next test-case deals with an optimization problem with a smaller Pareto-optimal area.

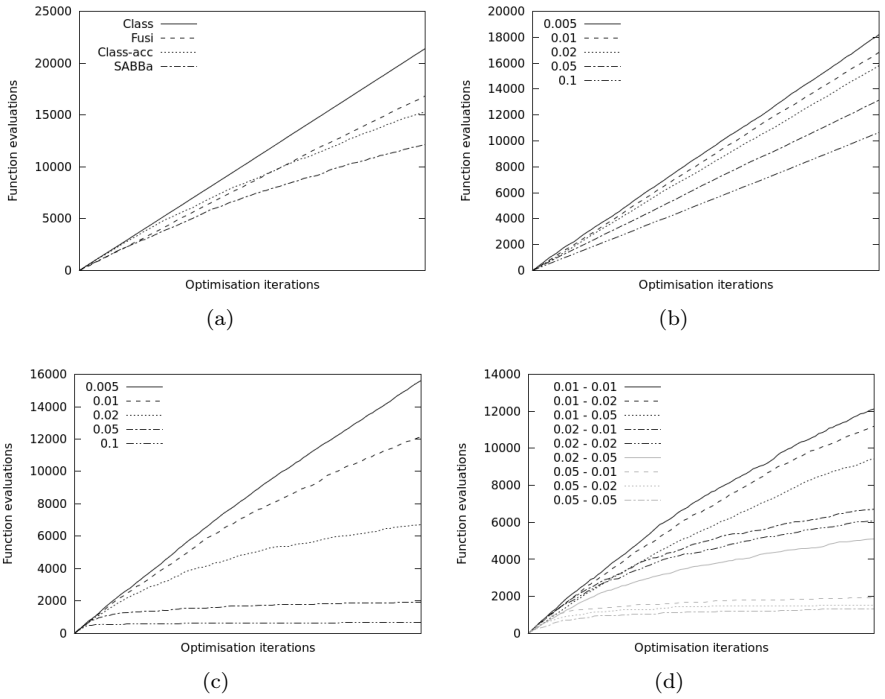


Figure 9: Test-case 1: a) number of function evaluations for each method of interest ($\bar{\boldsymbol{s}}_1 = \bar{\boldsymbol{s}}_2 = 0.01$), b) Fusi's approach varying $\bar{\boldsymbol{s}}_2$, c) SABBa framework varying $\bar{\boldsymbol{s}}_1$ with $\bar{\boldsymbol{s}}_2 = 0.01$; d) varying both $\bar{\boldsymbol{s}}_1$ and $\bar{\boldsymbol{s}}_2$.

5.3 Test-case 2: The Triangle problem

This problem is a bi-objective unconstrained optimization:

$$\begin{aligned} \text{minimize:} \quad & \boldsymbol{f}(\boldsymbol{x}) = \begin{pmatrix} \frac{x_1+x_2}{10} + |x_1 - x_2| \\ \frac{x_1}{5} + |x_2 - 2| \end{pmatrix} \\ \text{by changing:} \quad & (x_1, x_2) \in [0, 10]^2 \end{aligned} \tag{11}$$

The design space with the optimal set (represented with a thick black curve) and the objective space counterpart are represented in Fig. 11.

This small optimal area is well fit to study the behavior of SABBa when the optimizer converges toward the Pareto front. In practice, two variants are compared. At each iteration, 10 new random designs are given to SABBa: a) in $\mathcal{X} = [0, 10] \times [0, 10]$; b) in a domain $\mathcal{X}^k$ converging toward $[0, 2] \times [0, 2]$ throughout the iteration, with $\mathcal{X}^0 = \mathcal{X}$.

We aim at replicating the slow-down of the recursive Bounding-Box approach raised in [22], and at quantifying the impact of the SA strategy.

5.3.1 Test-case 2 a): No convergence of the designs

In this variant, new designs are chosen randomly within the $[0, 10]^2$ design space. The approximation of the Pareto front is very poor and highly inefficient, as one could expect. The result is pictured in Fig. 12.

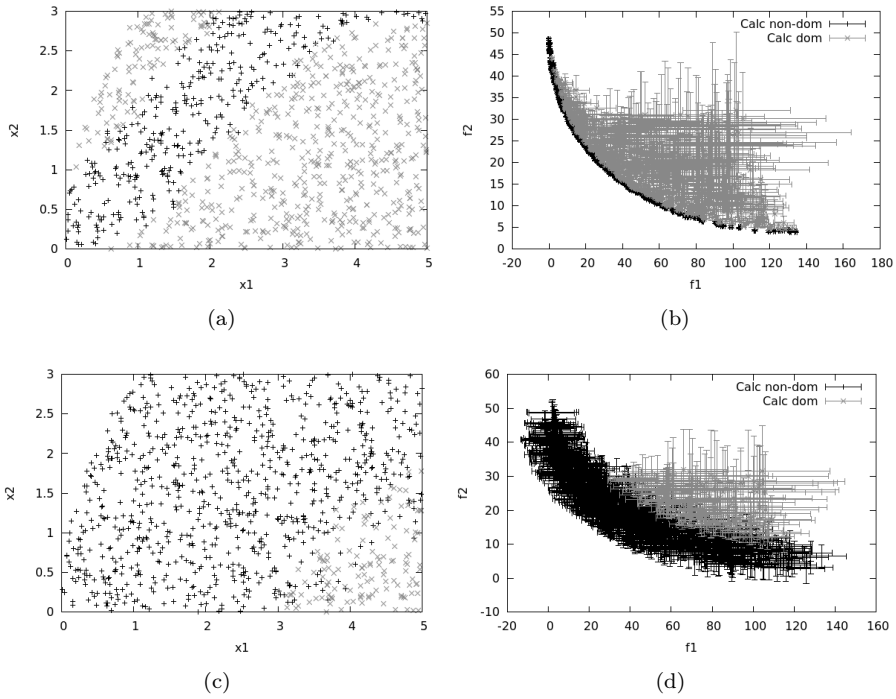


Figure 10: Impact of $\bar{\boldsymbol{s}}_2$: Outputs with $\bar{\boldsymbol{s}}_2 = 0.005$ in a) design space and b) objective space; Outputs with $\bar{\boldsymbol{s}}_2 = 0.1$ in c) design space and d) objective space.

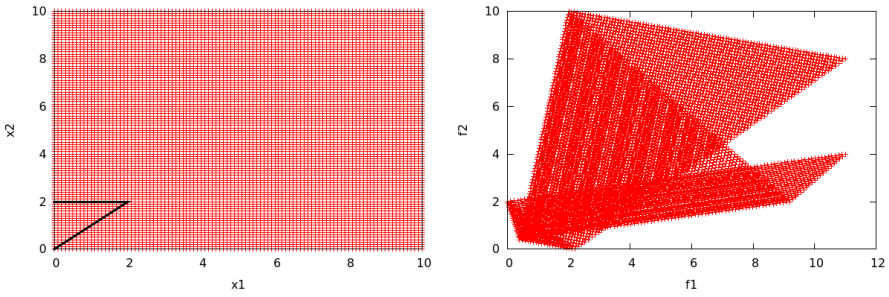


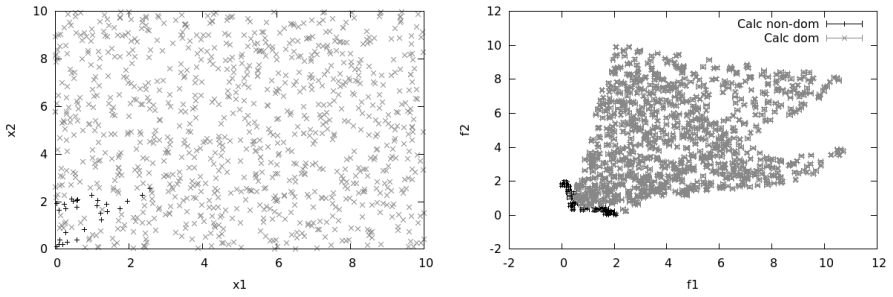
Figure 11: Test-case 2: a) acceptable design and Pareto optima (thick line), b) image in the objective space.

5.3.2 Test-case 2 b): Convergence of the designs

Here, new designs are randomly chosen in a domain that converges toward the optimal area. In practice, with j the optimization iteration, the new designs are drawn as follows, for i in $\{1, 2\}$,

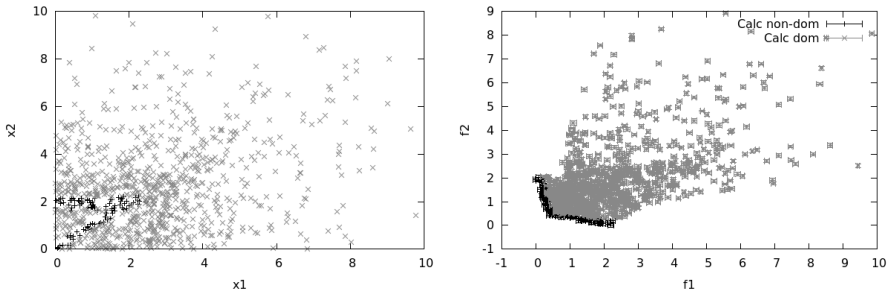
$$x_i = X\Big(2 + 8\exp\Big(-\frac{j}{45}\Big)\Big), \qquad \text{with } X \sim \mathcal{U}[0, 1].$$

This gives a better refinement of the Pareto front, as can be seen comparing Figs. 12 and 13.



(a) (b)

Figure 12: Test-case 2: *Classical approach*, non-converging domain for new designs: a) design space, b) objective space.



(a) (b)

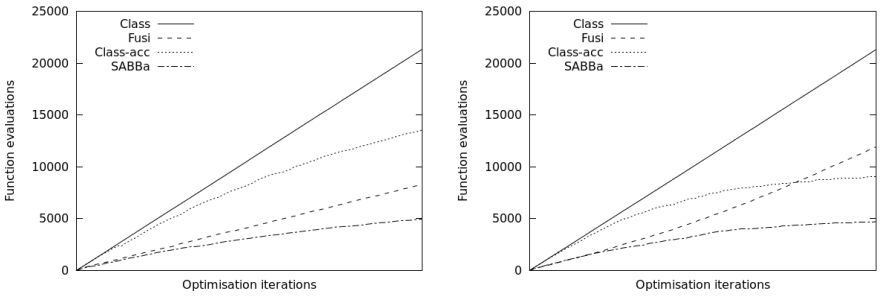
Figure 13: Test-case 2: *Classical approach*, converging domain for new designs: a) design space, b) objective space.

The number of function evaluations are compared in Fig. 14. As raised in [22], the convergence toward the optimal area induces a decreasing impact of the recursive strategy. The slope of the associated cost curve (dashed) slowly relapses to the slope of the classical one (solid) in Fig. 14b. This is due to the increasing ratio of non-dominated designs.

SABBa relies on the SA strategy to ultimately bypass function evaluations in the optimal area. In Fig. 14b, both surrogate-assited strategies (*Class-acc* and *SABBa*) benefit from the convergence to reach their cost plateau faster. Despite the increasing slope of the Fusi strategy alone, SABBa takes advantage of the convergence by constructing a locally highly refined surrogate model in the optimal area.

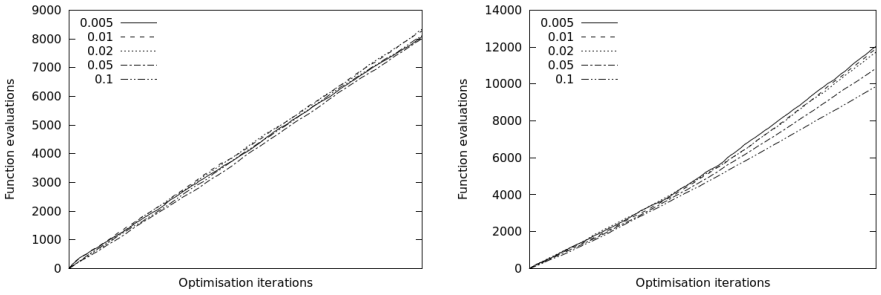
Fig. 14a can also be compared with Fig. 9a. Contrarily to the first test-case, Fig. 12 shows a small Pareto front. Many more boxes are dominated and the recursive strategy can generate higher computational gains.

Remark The increasing number of non-dominated design in the converging case also induces an increasing impact of the threshold $\bar{\boldsymbol{s}}_2$ throughout the optimization process. This is depicted in Figure 15, that compares the converging and non-converging cases.



(a) (b)

Figure 14: Test-case 2: number of function evaluations depending on the strategy for a) the non-converging case, b) the converging case.



(a) (b)

Figure 15: Test-case 2: number of function evaluations (*Fusi* strategy) depending on $\bar{\boldsymbol{s}}_2$ for a) the non-converging case, b) the converging case.

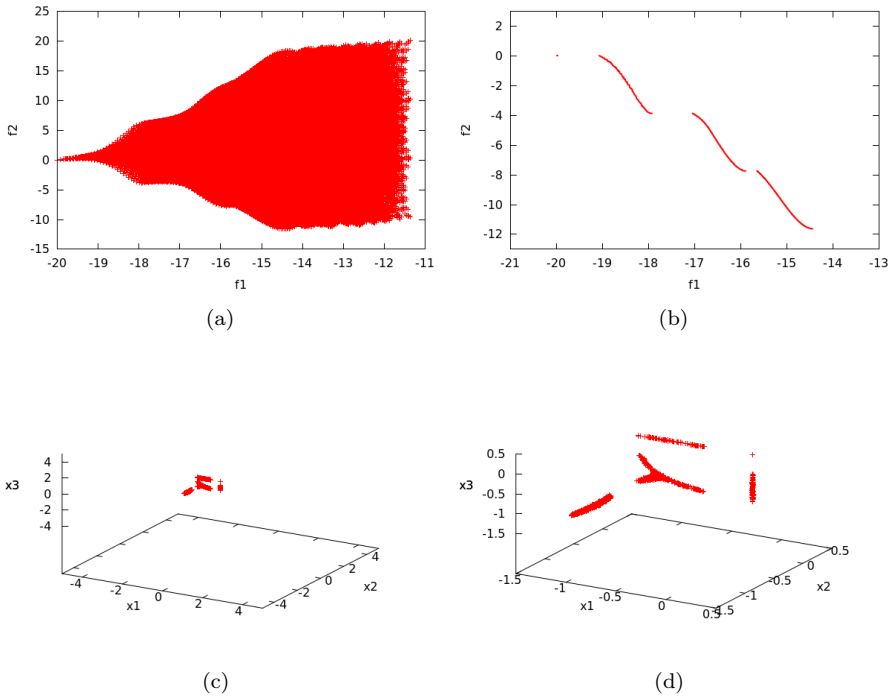
5.4 Test-case 3: The Kursawe problem

The Kursawe bi-objective optimization is performed here, in order to study the framework behavior on a more complex Pareto front shape. The problem is the following:

$$\begin{aligned} \text{minimize:} \quad & \boldsymbol{f}(\boldsymbol{x}) = \begin{pmatrix} \sum_{i=1}^{2} \left[-10 \exp\left(-0.2\sqrt{x_i^2 + x_{i+1}^2} \right) \right] \\ \sum_{i=1}^{3} \left[|x_i|^{0.8} + 5\sin(x_i^3) \right] \end{pmatrix} \\ \text{by changing:} \quad & (x_1, x_2, x_3) \in [-5, 5]^3 \end{aligned} \tag{12}$$

This problem is known to have a complex Pareto front with several discontinuities, associated with a small area of the design space. In the following, the convergence toward the optimal zone is forced, similarly to test-case 2 b). We aim at showing that the SABBa framework is not influenced by the complexity or dimensionality of the optimization problem. To tackle these issues, a suitable choice of the optimizer and the surrogate model is required.

The Pareto front associated to the problem is given in Fig. 16. It is highly discontinuous and should allow for a high impact of the recursive strategy as many designs are far from the Pareto front. The Pareto-optimal designs are represented in Fig. 16c in the $[-5, 5]^3$ domain and in Fig.



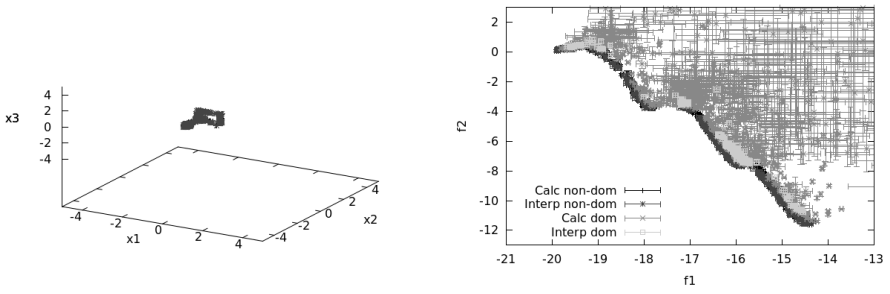
(a) (b)

(c) (d)

Figure 16: Test-case 3: a) image in the objective space, b) associated Pareto front, c) Pareto optima in the design space, d) in a close-up view.

16d in a close-up view. These designs form complex disjoint sets and their approximations will be qualitatively compared to Fig. 16c.

The SABBa framework is applied in the following with different thresholds. Fig. 17 gives the outputs obtained with a fine threshold. The results are highly comparable to Fig. 16, with only 28510 function evaluations over 48913 for the classical approach (**gain = 42%**).

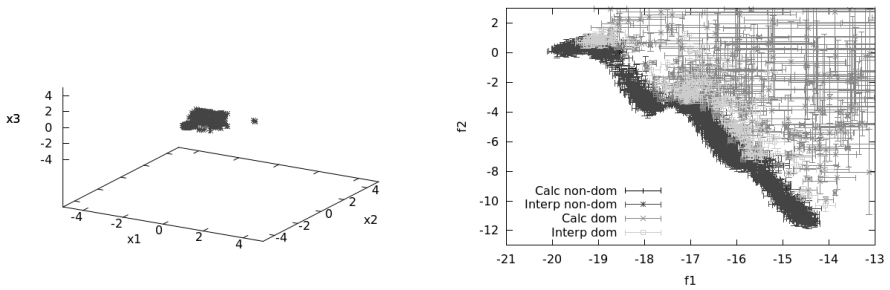


(a) (b)

Figure 17: Test-case 3: SABBa ouputs with fine thresholds in a) the design space, b) the objective space. Computational cost saving: 42%.

With larger thresholds, the optimal area is less accurately captured, as seen in Fig. 18a, and

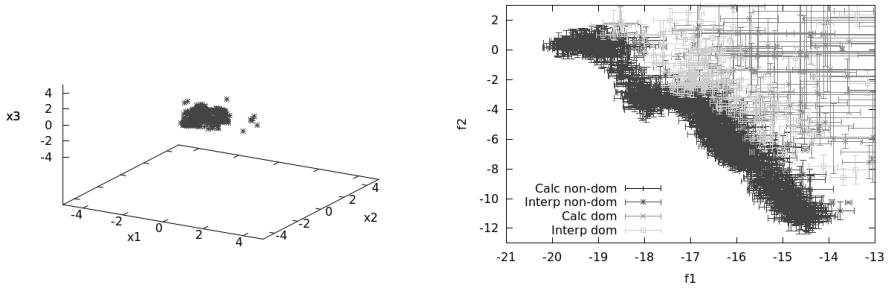
the associated approximated Pareto front is depicted with higher imprecision in Fig. 18b. Here, 8294 function evaluations were required (**gain = 83%**).



(a) (b)

Figure 18: Test-case 3: SABBa ouputs with moderate thresholds in a) the design space, b) the objective space. Computational cost saving: 83%.

With very coarse thresholds, the local optimal shape is not captured in Fig. 19. This however gives a first rough approximation of the optimal area with only a computational cost of 3546 evaluations (**gain = 93%**).



(a) (b)

Figure 19: Test-case 3: SABBa ouputs with coarse thresholds in a) the design space, b) the objective space. Computational cost saving: 93%.

The cost graphs are given in Fig. 20. Similarly to Figs. 14b and 15b, Figs. 20a and 20b reveal a decreasing impact of the recursive strategy alone and an increasing sensitivity to $\bar{s}_2$ throughout the optimization. Figure 20c and 20d show that the computational cost associated to SABBa is here highly influenced by the threshold $\bar{s}_1$, that controls the accuracy of the SA model.

SABBa yields a significant decrease in the computational cost. The user-defined thresholds heavily impact both the accuracy and the global computational cost. They should be chosen with care in real-life applications.

The efficiency of the SABBa framework is not directly correlated with the complexity of the optimization problem. The user should choose the most relevant optimization and surrogate modelling techniques for tackling high dimension, non-linearity or other difficulties. In this context, SABBa can bring consistent cost reduction through a fine allocation of the high-fidelity computations and robust use of the Surrogate-Assisting model.

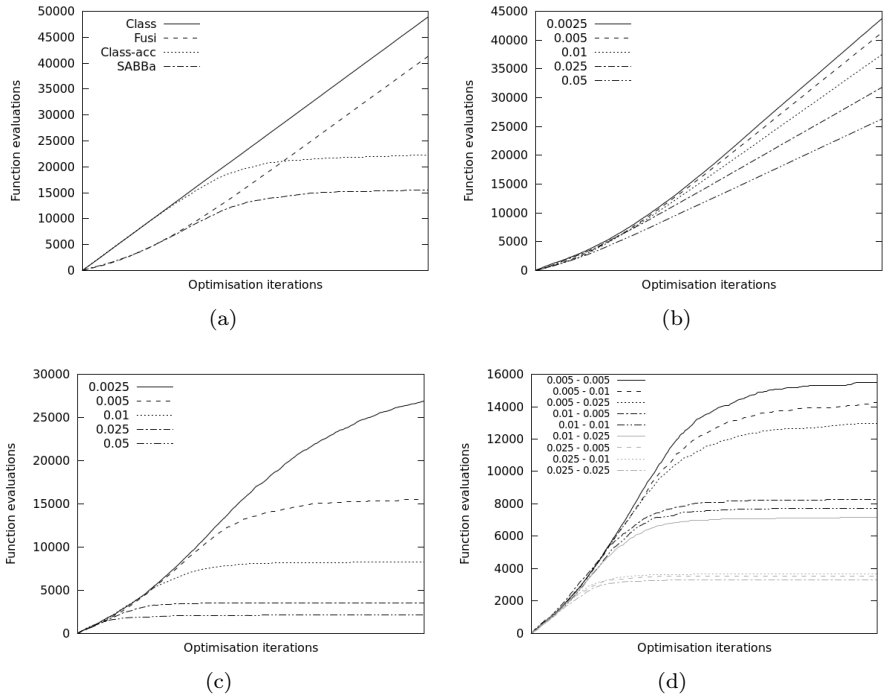


Figure 20: Test-case 3: a) number of function evaluations for each method of interest ($\bar{\boldsymbol{s}}_1 = \bar{\boldsymbol{s}}_2 = 0.01$), b) Fusi's approach varying $\bar{\boldsymbol{s}}_2$, c) SABBa framework varying $\bar{\boldsymbol{s}}_1$ with $\bar{\boldsymbol{s}}_2 = 0.01$; d) varying both $\bar{\boldsymbol{s}}_1$ and $\bar{\boldsymbol{s}}_2$.

6 Conclusive remarks

In this paper, we introduced a novel framework to efficiently deal with multi-objective (≤ 3) optimization problems with objective computations of tunable and refinable fidelity. This kind of problem could notably arise with iterative or sampling-based computations. The proposed strategy couples the recursive Bounding-Box approach for error-based objective functions developed in [22] with a Surrogate-Assisting strategy that aims at bypassing function evaluations. The intuitive idea is to use the Bounding-Box approach to quickly converge toward the optimal area through low-fidelity computations of the non-efficient designs and to simultaneously build the Surrogate-Assisting model, which refinement is driven by the optimization process. We first provided a mathematical proof of the robustness of the framework and of its convergence under some assumptions. Then, a numerical validation was carried out, in order to validate the convergence, quantify the computational gains and make explicit the main behaviors of the strategy.

The framework appears to yield a consistent cost-reduction compared to the Fusi [22] approach, which already performed better than the classical nested optimization approach. Accuracy of the outputs have been qualitatively analyzed with regards to the associated cost reduction. Finally, the influence of the user-defined thresholds has been investigated.

SABBa is based on several assumptions, such as the possibility to compute a conservative error approximation and the convergence of the optimization process. Plus, the objective estimation error is only represented through an uncorrelated bounded product of intervals. A high

correlation in the objective space could have a momentous impact on the domination computation, and thus on the computational cost. The uncorrelated assumption usually overestimates the true variability. Note also that the assumption of bounded error is critical for the computation of conservative errors, that ensure a proper convergence of the framework. A Gaussian-distributed error, unbounded, could be treated within the proposed framework in two different manners. The most straightforward approach would be to relax the conservative assumption by truncating the Gaussian distributions. One could also propose Gaussian objective approximations instead of uniform boxes. This would however require the choice of a suitable Pareto dominance rule.

Finally, the behavior of the framework when dealing with a high number of parameters has not been treated here. One would expect the Surrogate-Assisting model to need more training data before convergence, thus slowing down the whole process. Note that up to tens of dimensions, this issue is expected to be tackled with an appropriate choice of the surrogate model (*i.e.* Automatic Relevance Determination or additive kernel for Gaussian processes). Furthermore, the choice of the threshold $\boldsymbol{s}_1$ could be of particular importance and may allow to balance the computational burden with respect to the accuracy of the Pareto-optimal area.

Future developments will be directed towards the application of the proposed framework to a problem of uncertainty-based multi-objective optimization in an engineering context and to the management of constraint functions.

A Proof 1

Proof. The proof here is trivial. By definition,

$$\mathcal{X}^{\boldsymbol{f}}_{\widetilde{\mathcal{P}}}\Big(\big\{\boldsymbol{x}_i\big\}_{i=1}^N\Big) = \big\{\boldsymbol{x}_i,\ i \in [\![1,N]\!] \mid \text{non-domination condition}\big\} \subseteq \big\{\boldsymbol{x}_i,\ i \in [\![1,N]\!]\big\} \equiv \big\{\boldsymbol{x}_i\big\}_{i=1}^N.$$

The same can be said for the boxed Pareto optima. □ □

B Proof 2

Proof. For $\boldsymbol{y} \notin \mathcal{X}^{\boldsymbol{f}}_{\widetilde{\mathcal{P}}}\Big(\big\{\boldsymbol{x}_i\big\}_{i=1}^N\Big)$, let us assume that $\nexists \boldsymbol{y}' \in \mathcal{X}^{\boldsymbol{f}}_{\widetilde{\mathcal{P}}}\Big(\big\{\boldsymbol{x}_i\big\}_{i=1}^N\Big),\ \boldsymbol{y}' \succ \boldsymbol{y}$, then $\boldsymbol{y} \in \mathcal{X}^{\boldsymbol{f}}_{\widetilde{\mathcal{P}}}\Big(\mathcal{X}^{\boldsymbol{f}}_{\widetilde{\mathcal{P}}}\Big(\big\{\boldsymbol{x}_i\big\}_{i=1}^N\Big) \cup \boldsymbol{y}\Big) = \mathcal{X}^{\boldsymbol{f}}_{\widetilde{\mathcal{P}}}\Big(\big\{\boldsymbol{x}_i\big\}_{i=1}^N\Big)$ from Equation 6, which proves the first implication by contradiction. The second implication is immediate from the definition of $\mathcal{X}^{\boldsymbol{f}}_{\widetilde{\mathcal{P}}}$. □ □

C Proof 3

Proof. By using the explicit definition of the Pareto front as in Equation 4, the proof is immediate as Assumption 1 gives $\forall \boldsymbol{x} \in \mathcal{X}, \forall j \in [\![1,N]\!],\ f_j(\boldsymbol{x}) \in \big[\widetilde{f}_j(\boldsymbol{x}) - \overline{\varepsilon}_j(\boldsymbol{x}),\ \widetilde{f}_j(\boldsymbol{x}) + \overline{\varepsilon}_j(\boldsymbol{x})\big]$. Hence, $\forall \boldsymbol{x}_i \in \mathcal{X}^{\boldsymbol{f}}_{\widetilde{\mathcal{P}}}\Big(\big\{\boldsymbol{x}_i\big\}_{i=1}^N\Big)$:

$$\begin{aligned}\forall k \in [\![1,N]\!], \exists j \in [\![1,m]\!],\ \pm f_j(\boldsymbol{x}_i) &< \pm\, f_j(\boldsymbol{x}_k),\\ \pm\widetilde{f}_j(\boldsymbol{x}_i) - \overline{\varepsilon}_j(\boldsymbol{x}_i) &< \pm\, \widetilde{f}_j(\boldsymbol{x}_k) + \overline{\varepsilon}_j(\boldsymbol{x}_k).\end{aligned}$$

Therefore, $\boldsymbol{x}_i \in \mathcal{X}^{\boldsymbol{f}}_{\widetilde{\mathcal{P}}_{\mathcal{B}}}\Big(\big\{\big(\boldsymbol{x}_i, \widetilde{\boldsymbol{\varepsilon}}(\boldsymbol{x}_i)\big)\big\}_{i=1}^N\Big)$, which ends the proof. □ □

D Proof 4

Proof. For proving this, mathematical induction can be used.
Let us assume that $\exists l \in \mathbb{N}_+$, so that $\mathcal{X}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f}}\Big(\{\boldsymbol{x}_i\}_{i=1}^N\Big) \subseteq \widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k}$. Then $\forall \boldsymbol{x} \in \widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k}$:

- if $\boldsymbol{x} \in \mathcal{X}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f}}\Big(\{\boldsymbol{x}_i\}_{i=1}^N\Big)$, $\nexists \boldsymbol{y} \in \mathcal{X}$, so that $\boldsymbol{y} \succ \boldsymbol{x}$. Hence, $\nexists \boldsymbol{y} \in \widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k} \subseteq \mathcal{X}$, so that $\boldsymbol{y} \succ \boldsymbol{x}$, and therefore $\boldsymbol{x} \in \mathcal{X}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f}}(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k})$;
- if $\boldsymbol{x} \notin \mathcal{X}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f}}\Big(\{\boldsymbol{x}_i\}_{i=1}^N\Big)$, with Proposition 2, $\exists \boldsymbol{y} \in \mathcal{X}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f}}\Big(\{\boldsymbol{x}_i\}_{i=1}^N\Big) \subseteq \widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k}$, so that $\boldsymbol{y} \succ \boldsymbol{x}$, therefore, $\boldsymbol{x} \notin \mathcal{X}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f}}(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k})$.

which means that:

$$\mathcal{X}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f}}\Big(\{\boldsymbol{x}_i\}_{i=1}^N\Big) = \mathcal{X}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f}}(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k}). \tag{13}$$

Finally, Lemma 1 gives $\mathcal{X}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f}}(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k}) \subseteq \mathcal{X}_{\widetilde{\mathcal{P}}_{\mathcal{B}}}^{\boldsymbol{f}}\Big(\Big\{\Big(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}_i}^{\boldsymbol{f},k}, \widetilde{\varepsilon}^k\Big(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}_i}^{\boldsymbol{f},k}\Big)\Big)\Big\}_{i=1}^N\Big) = \widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k+1}$, which ends the inductive step of the proof, yielding $\mathcal{X}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f}}\Big(\{\boldsymbol{x}_i\}_{i=1}^N\Big) \subseteq \widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k+1}$.
Of course, $\mathcal{X}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f}}\Big(\{\boldsymbol{x}_i\}_{i=1}^N\Big) \subseteq \{\boldsymbol{x}_i\}_{i=1}^N = \widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},0}$, therefore, the mathematical induction proves that the robustness inclusion is verified. □ □

E Proof 5

Proof. The triangle inequality gives :

$$d_H\Big(\widetilde{\mathcal{P}}_c\big(\widetilde{\boldsymbol{f}}^{l_k}(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k})\big),\ \mathcal{P}_c\Big) \leq d_H\Big(\widetilde{\mathcal{P}}_c\big(\widetilde{\boldsymbol{f}}^{l_k}(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k})\big), \widetilde{\mathcal{P}}_c\big(\boldsymbol{f}(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k})\big)\Big) + d_H\Big(\widetilde{\mathcal{P}}_c\big(\boldsymbol{f}(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k})\big),\ \mathcal{P}_c\Big). \tag{14}$$

Assumption 1 implies that $\forall \boldsymbol{x} \in \widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k},\ d_\infty\big(\boldsymbol{f}(\boldsymbol{x}), \widetilde{\boldsymbol{f}}^{l_k}(\boldsymbol{x})\big) \leq \max\limits_j \widetilde{\boldsymbol{\varepsilon}}_j^{l_k}(\boldsymbol{x})$.
Now, let us suppose that $\exists \boldsymbol{a} \in \widetilde{\mathcal{P}}_c\big(\boldsymbol{f}(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k})\big)$, so that $d_\infty\Big(\boldsymbol{a}, \widetilde{\mathcal{P}}_c\big(\widetilde{\boldsymbol{f}}^{l_k}(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k})\big)\Big) > \max\limits_{(i,j)} \widetilde{\boldsymbol{\varepsilon}}_j^{l_k}\big(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}_i}^{\boldsymbol{f},k}\big)$.
This means that $\widetilde{\mathcal{P}}_c\big(\widetilde{\boldsymbol{f}}^{l_k}(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k})\big) \cap \big(\boldsymbol{a}, \widetilde{\boldsymbol{\varepsilon}}_{max}\big) = \emptyset$ with $\widetilde{\boldsymbol{\varepsilon}}_{max}$ being the m-dimensional vector where each component is equal to $\max\limits_{(i,j)} \widetilde{\boldsymbol{\varepsilon}}_j^{l_k}\big(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}_i}^{\boldsymbol{f},k}\big)$.
Therefore, from Def. 8, i) either $\exists i \in [\![1, N]\!]$, then $\big(\widetilde{\boldsymbol{f}}^{l_k}(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}_i}^{\boldsymbol{f},k}), 0\big) \underset{\mathcal{B}}{\succcurlyeq} \big(\boldsymbol{a}, \widetilde{\boldsymbol{\varepsilon}}_{max}\big)$ or ii) $\forall \boldsymbol{a}' \in \big(\boldsymbol{a}, \widetilde{\boldsymbol{\varepsilon}}_{max}\big), \nexists i \in [\![1, N]\!]$, so that $\widetilde{\boldsymbol{f}}^{l_k}(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}_i}^{\boldsymbol{f},k}) \succ \boldsymbol{a}'$.

- In the first case i), the dominance can be formulated as follows: $\exists j \in [\![1, N]\!]$, so that $\big(\widetilde{\boldsymbol{f}}^{l_k}(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}_j}^{\boldsymbol{f},k}), \widetilde{\boldsymbol{\varepsilon}}_{max}\big) \underset{\mathcal{B}}{\succcurlyeq} \big(\boldsymbol{a}, 0\big)$ and as $\boldsymbol{a} \in \widetilde{\mathcal{P}}_c\big(\boldsymbol{f}(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k})\big), \nexists i \in [\![1, N]\!], \boldsymbol{f}(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}_i}^{\boldsymbol{f},k}) \succcurlyeq \boldsymbol{a}$. Therefore, it can be inferred that $\nexists i \in [\![1, N]\!], \boldsymbol{f}(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}_i}^{\boldsymbol{f},k}) \in \big(\widetilde{\boldsymbol{f}}^{l_k}(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}_j}^{\boldsymbol{f},k}), \widetilde{\boldsymbol{\varepsilon}}_{max}\big)$. However, this would mean $\exists \boldsymbol{x} \in \widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k},\ d_\infty\big(\boldsymbol{f}(\boldsymbol{x}), \widetilde{\boldsymbol{f}}^{l_k}(\boldsymbol{x})\big) > \widetilde{\boldsymbol{\varepsilon}}_{max_i} > \max\limits_j \widetilde{\boldsymbol{\varepsilon}}_j^{l_k}(\boldsymbol{x})$, which is contradictory with Assumption 1.
- The second case ii) implies that $\exists \boldsymbol{b} \in \widetilde{\mathcal{P}}_c\big(\widetilde{\boldsymbol{f}}^{l_k}(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k})\big)$, so that $\big(\boldsymbol{a}, \widetilde{\boldsymbol{\varepsilon}}_{max}\big) \underset{\mathcal{B}}{\succcurlyeq} \big(\boldsymbol{b}, 0\big)$. However, $\exists j \in [\![1, N]\!]$, so that $\boldsymbol{f}(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}_j}^{\boldsymbol{f},k}) \succ \boldsymbol{a}$. Therefore, it follows $\big(\boldsymbol{f}(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}_j}^{\boldsymbol{f},k}), \widetilde{\boldsymbol{\varepsilon}}_{max}\big) \underset{\mathcal{B}}{\succcurlyeq} \big(\boldsymbol{b}, 0\big)$. As $\boldsymbol{b} \in \widetilde{\mathcal{P}}_c\big(\widetilde{\boldsymbol{f}}^{l_k}(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}}^{\boldsymbol{f},k})\big)$, from Def. 8, it follows that $\nexists i \in [\![1, N]\!]$, so that $\widetilde{\boldsymbol{f}}^{l_k}(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}_i}^{\boldsymbol{f},k}) \succcurlyeq \boldsymbol{b}$, hence, $\nexists i \in [\![1, N]\!], \widetilde{\boldsymbol{f}}^{l_k}(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}_i}^{\boldsymbol{f},k}) \in \big(\boldsymbol{f}(\widetilde{\mathcal{X}}_{\widetilde{\mathcal{P}}_j}^{\boldsymbol{f},k}), \widetilde{\boldsymbol{\varepsilon}}_{max}\big)$, which contradicts again Assumption 1.

Hence, we prove by contradiction that $\forall \boldsymbol{a} \in \widetilde{\mathcal{P}}_c\big(\boldsymbol{f}\big(\widetilde{\mathcal{X}}^{\boldsymbol{f},k}_{\widetilde{\mathcal{P}}}\big)\big)$, it follows that $d_\infty\Big(\boldsymbol{a}, \widetilde{\mathcal{P}}_c\big(\widetilde{\boldsymbol{f}}^{l_k}\big(\widetilde{\mathcal{X}}^{\boldsymbol{f},k}_{\widetilde{\mathcal{P}}}\big)\big)\Big) \leq \max\limits_{(i,j)} \widetilde{\varepsilon}^{l_k}_j\big(\widetilde{\mathcal{X}}^{\boldsymbol{f},k}_{\widetilde{\mathcal{P}}_i}\big)$. This statement holds also when inverting the Pareto front continuous sets (real and approximated), and this can proved in the same way. As a consequence, the Hausdorff distance can be written as follows:

$$d_H\Big(\widetilde{\mathcal{P}}_c\big(\widetilde{\boldsymbol{f}}^{l_k}\big(\widetilde{\mathcal{X}}^{\boldsymbol{f},k}_{\widetilde{\mathcal{P}}}\big)\big), \widetilde{\mathcal{P}}_c\big(\boldsymbol{f}\big(\widetilde{\mathcal{X}}^{\boldsymbol{f},k}_{\widetilde{\mathcal{P}}}\big)\big)\Big) \leq \max_{(i,j)} \widetilde{\varepsilon}^{l_k}_j\big(\widetilde{\mathcal{X}}^{\boldsymbol{f},k}_{\widetilde{\mathcal{P}}_i}\big).$$

Hence, Assumption 2 implies that:

$$\lim_{k\to\infty} d_H\Big(\widetilde{\mathcal{P}}_c\big(\widetilde{\boldsymbol{f}}^{l_k}\big(\widetilde{\mathcal{X}}^{\boldsymbol{f},k}_{\widetilde{\mathcal{P}}}\big)\big), \widetilde{\mathcal{P}}_c\big(\boldsymbol{f}\big(\widetilde{\mathcal{X}}^{\boldsymbol{f},k}_{\widetilde{\mathcal{P}}}\big)\big)\Big) = 0. \tag{15}$$

Let us focus now on the second part of the sum in Equation 14.

Of course, $\forall \boldsymbol{a} \in \widetilde{\mathcal{P}}_c\big(\boldsymbol{f}\big(\widetilde{\mathcal{X}}^{\boldsymbol{f},k}_{\widetilde{\mathcal{P}}}\big)\big), \exists \boldsymbol{a}' \in \mathcal{P}_c$, so that $\boldsymbol{a}' \succ \boldsymbol{a}$ (or $\boldsymbol{a}' = \boldsymbol{a}$). Moreover, $\forall \boldsymbol{b} \in \mathbb{R}^m$ such that $\exists \boldsymbol{a}' \in \mathcal{P},\ \boldsymbol{a}' \succcurlyeq \boldsymbol{b}$, Assumption 3 with Theorem 1 provides evidence that the recursive discrete efficient set converges toward the continuous real one and that this efficient set is included in $\widetilde{\mathcal{X}}^{\boldsymbol{f},k}_{\widetilde{\mathcal{P}}}$. In other words, $\forall k \in \mathbb{N}, \exists M \in \mathbb{N}^*, \exists i \in [\![1, M]\!], s.t. \forall j \in [\![1, m]\!], \big|a'_j - f_j\big(\widetilde{\mathcal{X}}^{\boldsymbol{f},k}_{\widetilde{\mathcal{P}}_i}\big)\big| < \big|a'_j - b_j\big|$. Thus, $\forall k \in \mathbb{N}, \exists M \in \mathbb{N}^*, \exists i \in [\![1, M]\!], \boldsymbol{f}\big(\widetilde{\mathcal{X}}^{\boldsymbol{f},k}_{\widetilde{\mathcal{P}}_i}\big) \succcurlyeq \boldsymbol{b}$. Hence, $\widetilde{\mathcal{P}}_c$ is always dominated by $\mathcal{P}_c$ and any element dominated by $\mathcal{P}_c$ is dominated by $\widetilde{\mathcal{P}}_c$ with a sufficient number of points. From Definition 8, it can be deduced that:

$$\lim_{N\to\infty} d_H\Big(\widetilde{\mathcal{P}}_c\big(\boldsymbol{f}\big(\widetilde{\mathcal{X}}^{\boldsymbol{f},k}_{\widetilde{\mathcal{P}}}\big)\big),\ \mathcal{P}_c\Big) = 0. \tag{16}$$

Finally, by combining Equations 14, 15 and 16, it comes:

$$\lim_{(N,l)\to(+\infty,+\infty)} d_H\Big(\widetilde{\mathcal{P}}_c\big(\widetilde{\boldsymbol{f}}^{l_k}\big(\widetilde{\mathcal{X}}^{\boldsymbol{f},k}_{\widetilde{\mathcal{P}}}\big)\big),\ \mathcal{P}_c\Big) = 0,$$

which ends the proof. □ □

F Proof 6

Proof. The proof is straightforward and comes from the following inequalities:
If $\widetilde{\varepsilon}^t_{SA}(\boldsymbol{x}_i) > \boldsymbol{s}_1$,

$$\big|\boldsymbol{f}_{opt}(\boldsymbol{x}_i) - \boldsymbol{f}(\boldsymbol{x}_i)\big| = \big|\widetilde{\boldsymbol{f}}^l(\boldsymbol{x}_i) - \boldsymbol{f}(\boldsymbol{x}_i)\big| \leq \widetilde{\boldsymbol{\varepsilon}}^l(\boldsymbol{x}_i).$$

Else,

$$\big|\boldsymbol{f}_{opt}(\boldsymbol{x}_i) - \boldsymbol{f}(\boldsymbol{x}_i)\big| = \big|\boldsymbol{f}^t_{SA}(\boldsymbol{x}_i) - \boldsymbol{f}(\boldsymbol{x}_i)\big| \leq \widetilde{\boldsymbol{\varepsilon}}^t_{SA}(\boldsymbol{x}_i)$$

which comes from Equation 3. □ □

References

[1] Yaochu Jin and J. Branke. Evolutionary optimization in uncertain environments-a survey. *IEEE Transactions on Evolutionary Computation*, 9(3):303–317, June 2005.

[2] Victor Picheny, David Ginsbourger, and Yann Richet. Noisy Expected Improvement and on-line computation time allocation for the optimization of simulators with tunable fidelity. working paper or preprint, June 2010.

[3] Saul Toscano-Palmerin and Peter I Frazier. Bayesian optimization with expensive integrands. *arXiv preprint arXiv:1803.08661*, 2018.

[4] Hisao Ishibuchi, Noritaka Tsukamoto, and Yusuke Nojima. Evolutionary many-objective optimization: A short review. In *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*, pages 2419–2426, June 2008.

[5] Kay Chen Tan and Chi Keong Goh. *Handling Uncertainties in Evolutionary Multi-Objective Optimization*, pages 262–292. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[6] C. Barrico and C. H. Antunes. Robustness analysis in multi-objective optimization using a degree of robustness concept. In *2006 IEEE International Conference on Evolutionary Computation*, pages 1887–1892, 2006.

[7] Mian Li, Shapour Azarm, and Vikrant Aute. A multi-objective genetic algorithm for robust design optimization. In *Proceedings of the 7th Annual Conference on Genetic and Evolutionary Computation*, GECCO '05, pages 771–778, New York, NY, USA, 2005. ACM.

[8] Kalyanmoy Deb and Himanshu Gupta. Introducing robustness in multi-objective optimization. *Evolutionary Computation*, 14(4):463–494, Nov 2006.

[9] H. Eskandari, C. D. Geiger, and R. Bird. Handling uncertainty in evolutionary multiobjective optimization: Spga. In *2007 IEEE Congress on Evolutionary Computation*, pages 4130–4137, Sept 2007.

[10] Evan J. Hughes. *Evolutionary Multi-objective Ranking with Uncertainty and Noise*, pages 329–343. Springer Berlin Heidelberg, Berlin, Heidelberg, 2001.

[11] J. E. Fieldsend and R. M. Everson. Multi-objective optimisation in the presence of uncertainty. In *2005 IEEE Congress on Evolutionary Computation*, volume 1, pages 243–250, Sept 2005.

[12] Jürgen Teich. *Pareto-Front Exploration with Uncertain Objectives*, pages 314–328. Springer Berlin Heidelberg, Berlin, Heidelberg, 2001.

[13] David L. J. Alexander, David W. Bulger, James M. Calvin, H. Edwin. Romeijn, and Ryan L. Sherriff. Approximate implementations of pure random search in the presence of noise. *Journal of Global Optimization*, 31(4):601–612, Apr 2005.

[14] Walter J. Gutjahr and Georg Ch. Pflug. Simulated annealing for noisy cost functions. *Journal of Global Optimization*, 8(1):1–13, Jan 1996.

[15] Antanas Žilinskas. On similarities between two models of global optimization: statistical models and radial basis functions. *Journal of Global Optimization*, 48(1):173–182, Sep 2010.

[16] D. w. Gong, Na na Qin, and Xiao yan Sun. Evolutionary algorithms for multi-objective optimization problems with interval parameters. In *2010 IEEE Fifth International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA)*, pages 411–420, Sept 2010.

[17] P. Limbourg and D. E. S. Aponte. An optimization algorithm for imprecise multi-objective problem functions. In *2005 IEEE Congress on Evolutionary Computation*, volume 1, pages 459–466 Vol.1, Sept 2005.

[18] G. L. Soares, F. G. Guimaraes, C. A. Maia, J. A. Vasconcelos, and L. Jaulin. Interval robust multi-objective evolutionary algorithm. In *2009 IEEE Congress on Evolutionary Computation*, pages 1637–1643, May 2009.

[19] Xiaoping Du. Unified uncertainty analysis by the first order reliability method. *Journal of Mechanical Design*, 130(9):091401–091401–10, Aug 2008.

[20] Philipp Limbourg. *Multi-objective Optimization of Problems with Epistemic Uncertainty*, pages 413–427. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[21] Miha Mlakar, Tea Tusar, and Bogdan Filipic. Comparing solutions under uncertainty in multiobjective optimization. *Mathematical Problems in Engineering*, 2014:1–10, 2014.

[22] Francesca Fusi and Pietro Marco Congedo. An adaptive strategy on the error of the objective functions for uncertainty-based derivative-free optimization. *Journal of Computational Physics*, 309:241–266, February 2016.

[23] Virginia Torzcon and Michael W Trosset. Using approximations to accelerate engineering design optimization. In *7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization: A Collection of Technical Papers, Part 2*, pages 738–748. American Institute of Aeronautics and Astronautics, 1998.

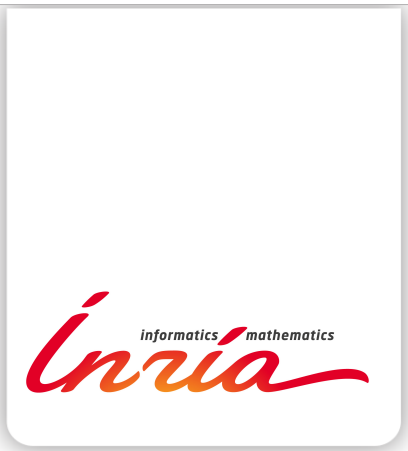
[24] Yaochu Jin, M. Olhofer, and B. Sendhoff. A framework for evolutionary optimization with approximate fitness functions. *IEEE Transactions on Evolutionary Computation*, 6(5):481–494, Oct 2002.

[25] Yaochu Jin. Surrogate-assisted evolutionary computation: Recent advances and future challenges. *Swarm and Evolutionary Computation*, 1(2):61 – 70, 2011.

[26] Michael Emmerich, Alexios Giotis, Mutlu Özdemir, Thomas Bäck, and Kyriakos Giannakoglou. *Metamodel—Assisted Evolution Strategies*, pages 361–370. Springer Berlin Heidelberg, Berlin, Heidelberg, 2002.

[27] M. T. M. Emmerich, K. C. Giannakoglou, and B. Naujoks. Single- and multiobjective evolutionary optimization assisted by gaussian random field metamodels. *IEEE Transactions on Evolutionary Computation*, 10(4):421–439, Aug 2006.

[28] D. Buche, N. N. Schraudolph, and P. Koumoutsakos. Accelerating evolutionary algorithms with gaussian process fitness function models. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 35(2):183–194, May 2005.



RESEARCH CENTRE
SACLAY – ÎLE-DE-FRANCE

1 rue Honoré d'Estienne d'Orves
Bâtiment Alan Turing
Campus de l'École Polytechnique
91120 Palaiseau

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399