

Upper Trust Bound Feasibility Criterion for Mixed Constrained Bayesian Optimization with Application to Aircraft Design

Rémy Priem, Nathalie Bartoli, Youssef Diouane, Alessandro Sgueglia

▶ **To cite this version:**

Rémy Priem, Nathalie Bartoli, Youssef Diouane, Alessandro Sgueglia. Upper Trust Bound Feasibility Criterion for Mixed Constrained Bayesian Optimization with Application to Aircraft Design. Aerospace Science and Technology, In press, 10.1016/j.ast.2020.105980 . hal-02569720

HAL Id: hal-02569720
https://hal.science/hal-02569720

Submitted on 11 May 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Upper Trust Bound Feasibility Criterion for Mixed Constrained Bayesian Optimization with Application to Aircraft Design

R. Priem[a,b], N. Bartoli[a], Y. Diouane[b], A. Sgueglia[a,b]

[a] *ONERA, DTIS, Université de Toulouse, Toulouse, France*
[b] *ISAE-SUPAERO, Université de Toulouse, Toulouse, 31055 Cedex 4, France*

Abstract

Bayesian optimization methods have been successfully applied to *black box* optimization problems that are *expensive to evaluate*. In this paper, we adapt the so-called *super efficient global optimization* algorithm to solve more accurately mixed constrained problems. The proposed approach handles constraints by means of *upper trust bound*, the latter encourages exploration of the feasible domain by combining the mean prediction and the associated uncertainty function given by the Gaussian processes. On top of that, a refinement procedure, based on a learning rate criterion, is introduced to enhance the exploitation and exploration trade-off. We show the good potential of the approach on a set of numerical experiments. Finally, we present an application to conceptual aircraft configuration upon which we show the superiority of the proposed approach compared to a set of the state-of-the-art black box optimization solvers.

Keywords: Global Optimization, Mixed Constrained Optimization, Black box optimization, Bayesian Optimization, Gaussian Process.

1. Introduction

In this paper, we are interested in the following mixed constrained optimization problem

$$\min_{\boldsymbol{x}\in\Omega} \{f(\boldsymbol{x}) \ \text{ s.t. } \ \boldsymbol{g}(\boldsymbol{x}) \geq 0 \text{ and } \boldsymbol{h}(\boldsymbol{x}) = 0\} \tag{1}$$

where $f : \mathbb{R}^d \mapsto \mathbb{R}$ is the objective function, $\boldsymbol{g} : \mathbb{R}^d \mapsto \mathbb{R}^m$ gives the inequality constraints, and $\boldsymbol{h} : \mathbb{R}^d \mapsto \mathbb{R}^p$ returns the equality constraints. The design space $\Omega \subset \mathbb{R}^d$ is a bounded domain. The functions f, $\boldsymbol{g}$, and $\boldsymbol{h}$ are typically simulations which possess no exploitable properties such as derivatives (i.e., black box). They take also a long time to evaluate. In some cases, these functions may be multimodal and the resulting feasible domain more complex to define.

Many practical optimization problems are in a black box form, expensive to evaluate and present mixed multimodal constraints. For instance, for *multidisciplinary design optimization* (MDO) problems [1], one has to conceive the best product regarding specific performances. The design domain is restricted by several mixed constraints coming from the requirements, the different disciplines involved (e.g., structure, aerodynamic and propulsion in aircraft design) and their interconnections. In this context, *Bayesian optimization* (BO) is a powerful strategy for solving problem (1).

Most of the work on BO [2–4] focuses on unconstrained black box optimization problems using a sequential enrichment of surrogate models. In fact, using *Gaussian processes* (GPs) [5, 6] to define response surfaces, the sequential enrichment is performed by maximizing a given acquisition function [2, 3]. The latter is meant to model a compromise between exploration of new zones in the design space and exploitation (i.e. minimization) of the GPs.

Extensions of the BO framework have been developed to handle constraints [2, 3, 7–9]. In this case, both the objective and the constraints functions are modeled with GPs, and an optimization sub-problem based on the infill criterion leads the enrichment process. Existing constrained BO methods can be split into two main categories. The first one, which is the largest, addresses only inequality constrained problems [2, 3, 7–9] and the references therein. We note that most of the existing works recommend to handle equality constraints by transformation of the initial optimization problem. For instance, equality constraints of the type $\boldsymbol{h}(\boldsymbol{x}) = 0$ are changed into two inequality constraints of the type $\boldsymbol{h}(\boldsymbol{x}) \geq 0$ and $-\boldsymbol{h}(\boldsymbol{x}) \geq 0$. In general, such transformation turns out to be harmful as it increases the number of constraints and also introduces antagonist requirements

leading to constraint qualification issues [10]. Alternatively, other approaches proposed to address mixed constrained problems without any transformation [11, 12].

ALBO [12] is the state-of-the-art solver in BO to handle mixed constrained problems. It combines an unconstrained BO framework with the classical *augmented Lagrangian* (AL) framework [10]. ALBO was originally designed for the equality constraints problems [13] and then extended to inequality constraints by means of the slack variables. The ALBO procedure is the same as the AL framework except that the minimization of the AL function is replaced by the maximization of an acquisition function. The new acquisition function is not given explicitly but only through an estimation method. Despite of the introduced effort in reducing the computational cost of the acquisition function estimation, the ALBO process is still not adapted to solve large scale problems in a reasonable time.

The *super efficient global optimization* (SEGO) framework [11] is an extension of the well-known unconstrained efficient global optimization framework [14] to handle mixed constrained optimization problems. The SEGO enrichment process is led by a constrained optimization sub-problem where the objective function is given by an acquisition function and the constraints by the GPs mean predictions of the constraints functions (without inclusion of the uncertainties provided by the GPs). By doing so, the constraints can be badly approximated by the GPs and the optimization can be misled especially when constraints are hard to approximate. The main advantage of the SEGO framework, compared to others, is related to the fact that it scales well when solving large scale constrained optimization problems, e.g., Bouhlel et al. [15].

To tackle the issue of badly modeled constraints, including the uncertainties (provided by the GP models) has been shown to be very useful [16–18]. For instance, the authors in [16, 17] introduced a scalar fixed *upper trust bound* (UTB) to handle the constraints. Their proposed approach was designed to allow the exploration of a larger feasible domain (but relaxed) rather than being restricted to a small feasible one. Recently, unlike in [16, 17] where only a fixed

scalar UTB was investigated, Priem et al. [18] used a dynamic adaptive strategy for updating the UTB during the optimization process. The main idea was to include uncertainties on GPs but only during specific stages of the optimization procedure. This led to a good compromise between exploration (of the design space) and exploitation (i.e., minimization of the objective function). It provided encouraging results on difficult optimization toy problem. We stress that all the approaches [16–18] were designed to handle only inequality constraints and thus are not adapted to solve mixed constrained optimization problems of the form (1). We note also that all the works [16–18] were only validated using toy problems. Thus, confirming the potential of all these approaches using extensive and practical numerical tests can be very useful.

In this paper, in the context of the SEGO framework, we propose to improve the existing constraints handling strategies [16, 18] by (a) including equality constraints, (b) using a better adaptive mechanism for the update of the UTB during the optimization process, and (c) performing extensive and practical numerical tests on a large test set of problems. In our proposed approach, the UTB is controlled using a learning rate vector that helps in managing the trade-off between exploration of badly modeled domain and exploitation of the known feasible domain predicted by the surrogate models. In fact, the extended version of SEGO for equality constraints using the *upper trust bound*, called SEGO-UTB, is shown to ensure a better exploration of the entire feasible domain. Its superiority, compared to SEGO and other solvers, is confirmed on 29 mixed constrained problems using different test strategies including data profiles. Finally, SEGO-UTB is applied to solve an MDO problem where the goal is to optimize a "tube & wing" hybrid aircraft configuration with a distributed electric propulsion [19].

The outline of the paper is as follows. In Section 2, a detailed review of the constrained Bayesian optimization framework is given. The adaptive UTB as well as different constraints learning rate strategies are given in Section 3. Section 4 presents our academical tests. The MDO test case is commented in Section 5. Conclusions and perspectives are finally drawn in Section 6.

2. Bayesian optimization and the SEGO framework

2.1. Bayesian optimization framework

Starting from an initial *design of experiments* (DoE) using a first set of l sample points chosen in the design domain Ω, constrained BO framework builds surrogate models using GPs [5, 6] of the objective f, inequality $\boldsymbol{g}$ and equality $\boldsymbol{h}$ constraints functions. The surrogate models are then iteratively enriched in order to locate the optimum of the constrained optimization problem. The search strategy balances the exploration of the design space Ω and the exploitation of the surrogate models by solving a maximization mixed-constrained sub-problem. The objective function of the sub-problem is expressed by an acquisition function while the constraints use GPs to replace $\boldsymbol{g}$ and $\boldsymbol{h}$. Solving the sub-problem is assumed to be computationally inexpensive as one uses only GPs information. Iteratively, the solution of the sub-problem is evaluated on f, $\boldsymbol{g}$, $\boldsymbol{h}$, and added to the respective DoE. The same process is repeated until a maximum number of iterations is reached. The main steps of the BO framework, when applied to problem (1), are summarized by Algorithm 1. The next two subsections describe

Algorithm 1 The Bayesian optimization framework.

input : Objective and constraints functions, initial DoEs for objective and constraints, a maximum number of iterations max_nb_it

1: **for** $l = 0$ **to** max_nb_it - 1 **do**
2: Build the surrogate models using GPs
3: Find $\boldsymbol{x}^{(l+1)}$ a solution of the enrichment maximization sub-problem
4: Evaluate the objective and constraints functions at $\boldsymbol{x}^{(l+1)}$
5: Update the DoE
6: **end for**

output : The best point found in the DoE

the information provided by the GPs as well as the maximization sub-problem choices within the constrained BO framework.

2.2. Gaussian Process

Scalar output GPs [6] are fully defined by a mean function μ and a standard deviation function σ. The mean function describes the global behaviour of the GP whereas the standard deviation function depicts the GP uncertainty of each sample on the entire domain.

A description of GPs can be as follows. Let $s : \mathbb{R}^d \mapsto \mathbb{R}$ be a scalar function for which a GP is built using a DoE of l points $\mathcal{D}_s^{(l)} = \{\boldsymbol{x}^{(k)}, y_s^{(k)}\}_{k=1,\dots,l}$ where $\boldsymbol{x}^{(k)} \in \Omega$ and $y_s^{(k)} = s(\boldsymbol{x}^{(k)}) \in \mathbb{R}$. For clarity reasons, in the context of our optimization problem (1), s can represent the objective function (i.e., $s = f$) or a given component constraint function (i.e., $s = g_j$ or h_j for a given constraint component j).

The GP model related to s using l sample points is a family of functions defined by a mean function $\mu_s^{(l)}$ and a standard deviation $\sigma_s^{(l)}$. Namely, at each point $\boldsymbol{x}$ of the bounded domain Ω, the GP of s is defined with a multivariate Gaussian distribution $\mathcal{N}(\mu_s^{(l)}(\boldsymbol{x}), \sigma_s^{(l)}(\boldsymbol{x}))$.

Note that the mean $\mu_s^{(l)}$ and $\sigma_s^{(l)}$ are computed thanks to a correlation function chosen by the user. In fact, the definition of the correlation function depends on a set of hyper-parameters that are in general estimated by maximizing a likelihood function. Unfortunately, such maximization can be computationally challenging for large scale functions or with a large DoE. Practical approaches to estimate the hyper-parameters can be found in [15, 20].

2.3. The enrichment optimization sub-problem

The BO framework combines the surrogate models provided by GPs and the enrichment strategy driven by the maximization of the sub-problem. In fact, for a given iteration l, the GPs of the objective f and each component of the constraints g, h are built using the current DoE ($\mathcal{D}_f^{(l)}$, $\mathcal{D}_{g_i}^{(l)}$, $\mathcal{D}_{h_j}^{(l)}$) for $i = 1, \dots, m$ and $j = 1, \dots, p$. The corresponding GPs mean and standard deviation functions are $\mu_f^{(l)}$, $\mu_{g_i}^{(l)}$, $\mu_{h_j}^{(l)}$, $\sigma_f^{(l)}$, $\sigma_{g_i}^{(l)}$ and $\sigma_{h_j}^{(l)}$ for $i = 1, \dots, m$ and $j = 1, \dots, p$. For clarity reasons, we will use the following vector notations for inequality constraints $\boldsymbol{g}$, i.e., $\boldsymbol{\mu_g}^{(l)} = [\mu_{g_1}^{(l)}, \dots, \mu_{g_m}^{(l)}]^\top \in \mathbb{R}^m$, $\boldsymbol{\sigma_g}^{(l)} = [\mu_{g_1}^{(l)}, \dots, \sigma_{g_m}^{(l)}]^\top \in \mathbb{R}^m$, and similarly for

equality constraints $\boldsymbol{h}$, i.e., $\boldsymbol{\mu}_{\boldsymbol{h}}^{(l)} = [\mu_{h_1}^{(l)}, \ldots, \mu_{h_p}^{(l)}]^\top \in \mathbb{R}^p$, $\boldsymbol{\sigma}_{\boldsymbol{h}}^{(l)} = [\mu_{h_1}^{(l)}, \ldots, \sigma_{h_p}^{(l)}]^\top \in \mathbb{R}^p$.

Originally, an unconstrained BO framework is led by an acquisition function $\alpha_f^{(l)}$ modeling the trade-off between exploration of new areas in the design space (i.e., areas with high value of $\sigma_f^{(l)}$) and exploitation (i.e., minimization of $\mu_f^{(l)}$). The most promising point is given by maximization of the acquisition function:

$$\boldsymbol{x}^{(l+1)} = \arg\max_{\boldsymbol{x}\in\Omega} \alpha_f^{(l)}(\boldsymbol{x}).$$

The acquisition functions, that have been developed, are either explicit or implicit [2–4, 21]. The computational cost of the implicit ones forbids their use for large scale optimization problems.

In the context of constrained optimization problems, for a given outer iteration l, the BO framework has been extended in two following ways. The first way is by using a merit type function $\alpha_m^{(l)} : \mathbb{R}^d \mapsto \mathbb{R}$ where one combines the objective and the constraints. The new enrichment point is thus computed by maximizing the merit function $\alpha_m^{(l)}$ on the design space Ω, i.e.,

$$\boldsymbol{x}^{(l+1)} = \arg\max_{\boldsymbol{x}\in\Omega} \alpha_m^{(l)}(\boldsymbol{x}).$$

Several methods based of this merit function have been proposed in the literature [2, 3, 8, 9], ALBO is belonging to this class of methods.

The second way of handling constraints, in the context of BO, consists in solving a mixed constrained maximization sub-problem, i.e.,

$$\boldsymbol{x}^{(l+1)} = \arg\max_{\boldsymbol{x}\in\Omega_{\boldsymbol{g}}^{(l)}\cap\Omega_{\boldsymbol{h}}^{(l)}} \alpha_f^{(l)}(\boldsymbol{x}), \tag{2}$$

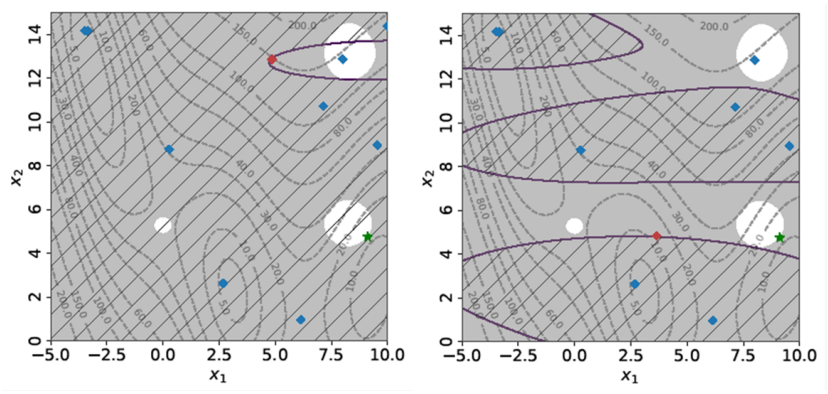
where $\alpha_f^{(l)} : \mathbb{R}^d \mapsto \mathbb{R}$ is a given acquisition function related to the objective function f (similarly to the unconstrained case), $\Omega_{\boldsymbol{g}}^{(l)}$ and $\Omega_{\boldsymbol{h}}^{(l)}$ are respectively the approximated feasible domains defined by the feasibility criteria $\boldsymbol{\alpha}_{\boldsymbol{g}}^{(l)} : \mathbb{R}^d \mapsto \mathbb{R}^m$ and $\boldsymbol{\alpha}_{\boldsymbol{h}}^{(l)} : \mathbb{R}^d \mapsto \mathbb{R}^p$. We note that the feasibility criteria $\boldsymbol{\alpha}_{\boldsymbol{g}}^{(l)}$ and $\boldsymbol{\alpha}_{\boldsymbol{h}}^{(l)}$ are not necessarily of the same form as $\boldsymbol{g}$ and $\boldsymbol{h}$. For instance, equality constraints can be expressed as inequality approximated constraints, in this case, the approximated

feasible domain $\Omega_{\boldsymbol{h}}^{(l)}$ related to the constraints $\boldsymbol{h}$ will be of the form $\{\boldsymbol{x} \in \Omega \ ; \ \boldsymbol{\alpha}_{\boldsymbol{h}}^{(l)}(\boldsymbol{x}) \geq 0\}$.

To the best of our knowledge, a multitude of existing approaches [2, 3, 11, 16–18] uses a constrained maximization sub-problem of the form (2), but only SEGO [11] was designed to solve mixed constrained optimization problems. In this context, SEGO sets the feasibility criteria functions $\boldsymbol{\alpha}_{\boldsymbol{g}}^{(l)}$ and $\boldsymbol{\alpha}_{\boldsymbol{h}}^{(l)}$ to be equal to the prediction of the GP models of the constraints $\boldsymbol{\mu}_{\boldsymbol{g}}^{(l)}$ and $\boldsymbol{\mu}_{\boldsymbol{h}}^{(l)}$ on the following way $\Omega_{\boldsymbol{g}}^{(l)} = \{\boldsymbol{x} \in \Omega \ ; \ \boldsymbol{\mu}_{\boldsymbol{g}}^{(l)}(x) \geq 0\}$ and $\Omega_{\boldsymbol{h}}^{(l)} = \{\boldsymbol{x} \in \Omega \ ; \ \boldsymbol{\mu}_{\boldsymbol{h}}(x) = 0\}$. For SEGO, by using only the mean functions of the GPs to model the constraints, all the functions involved in the mixed constrained maximization sub-problem are explicit and computationally inexpensive. On the contrary, the implicit methods, where typically Monte-Carlo estimators are used for each evaluation of the acquisition functions, are expensive to compute.

Despite the good results that SEGO has shown [11, 21], the use of only the mean functions of the GPs to model the constraints of the sub-problem (2) can mislead the optimization process and impact it badly. In fact, during its early stage, the DoE is still poor as it does not provide enough information to build accurate GPs. Due to the large uncertainties on the GPs defining $\boldsymbol{g}$ and $\boldsymbol{h}$, using only the mean functions $\boldsymbol{\mu}_{\boldsymbol{g}}^{(l)}$ and $\boldsymbol{\mu}_{\boldsymbol{h}}^{(l)}$, may consider that most of the design space is unfeasible. Hence a large part of the feasible domain may not be explored. In this case, the enrichment process gets very local and may ignore other feasible areas of the design space.

As example, in Figure 1a, we show the feasible domain (the true and the predicted one) and the contour plots of the objective function for the modified Branin problem [22]. The true unfeasible domain is represented with the grey color (the feasible domain is formed by three disjoint balls) and the predicted unfeasible domain at the first iteration is represented by the hatched area. One can clearly see that the feasible area, as predicted by SEGO, is not covering two of the three true feasible domains (white areas). Due to that, during the maximization sub-problem, SEGO will only provide enrichment points within



(a) SEGO (b) SEGO-UTB

Figure 1: A representation of the SEGO/SEGO-UTB feasible domain (as predicted at a given iteration) for the modified Branin problem. The hatched area is the SEGO/SEGO-UTB unfeasible domain, the grey area shows the true unfeasible domain and the dashed curves are the contour plots of the objective function. The blue squares represent the current DoE whereas the green star indicates the global minimum of the problem. The red square is the new point to add in the DoE.

the predicted feasible domain. Consequently, the true feasible domain, where the global optimum is located, will never be explored by SEGO. To overcome this issue, we will introduce in the next section a new feasibility criterion that explores more efficiently the design domain whenever the provided GPs for the constraints are inaccurate.

3. Mixed constrained BO by using upper trust bounds

3.1. On the use of upper trust bounds for constraints estimation

At the end of Section 2, we explain that SEGO may mislead the optimization process. This is due to a lack of accuracy of the constraints GPs, as it only uses the mean of the predicted values by the GPs without taking into account the level of accuracy associated to such estimation.

In the context of unconstrained BO optimization, a combination of both functions $\mu_s^{(l)}$ and $\sigma_s^{(l)}$ was used to provide an *upper confidence bound* (UCB)

[23] on the acquisition function, and shown to lead to a more robust model approximation for the objective function (up to a confidence level). Similarly, we mimic the UCB strategy to better estimate the constraints within the SEGO framework. The key idea is as follows: for a given scalar function s, using a learning rate $\tau_s^{(l)} \geq 0$, the functions $\mu_s^{(l)} \pm \tau_s^{(l)}\sigma_s^{(l)}$ approximate the targeted function s with a trust level that is related to $\tau_s^{(l)}$. For instance, when $\tau_s^{(l)} = 3$ the trust interval can be expressed with 99% confidence for a single point x sampled in Ω. Outside of this zone, the value cannot be trusted as a reliable sample of the GP. Based on this observation, we will try to model the constraints ($\boldsymbol{h}$ and $\boldsymbol{g}$) in a more robust way by introducing an *upper trust bound* (UTB) on the constraints. The proposed UTB mechanisms depend on the nature of the regarded constraint (i.e., equality or inequality).

Concerning the inequality constraints $\boldsymbol{g}$, the UTB mechanism, by means of $\boldsymbol{\alpha}_{\boldsymbol{g}}^{(l)}$, tries to relax the predicted feasible domain so that it includes the true feasible domain with a high probability. The given trust level is governed by $\boldsymbol{\tau}_{\boldsymbol{g}}^{(l)} = [\tau_{g_1}^{(l)}, \ldots, \tau_{g_m}^{(l)}]^\top \in \mathbb{R}_+^m$, namely, one has

$$\Omega_g^{(l)} \quad = \quad \left\{ \boldsymbol{x} \in \Omega \ ; \ \boldsymbol{\mu}_{\boldsymbol{g}}^{(l)}(\boldsymbol{x}) + \boldsymbol{\tau}_{\boldsymbol{g}}^{(l)} \odot \boldsymbol{\sigma}_{\boldsymbol{g}}^{(l)}(\boldsymbol{x}) \geq 0 \right\} \tag{3}$$

where the operator "$\odot$" denotes the element-wise multiplication.

Figure 1b shows the trusted feasible zone at 99% for a single point $\boldsymbol{x}$ sampled in Ω (that can be found with a learning rate equals to 3) of the modified Branin problem. Compared to SEGO, see Figure 1a, the use of UTB, in the constraints formulation of the SEGO maximization sub-problem, leads to a bigger predicted feasible domain that, here, includes all the true feasible areas. This allows a better exploration of the feasible domain and hence finding the global optimum.

For the equality constraints $\boldsymbol{h}$, the UTB feasibility criterion is less straightforward. It expresses the best constraints approximation (with the smallest violation) within the trusted domain delimited by the vectors $\boldsymbol{\mu}_{\boldsymbol{h}}^{(l)}(\boldsymbol{x}) - \boldsymbol{\tau}_{\boldsymbol{h}}^{(l)} \odot \boldsymbol{\sigma}_{\boldsymbol{h}}^{(l)}$ and $\boldsymbol{\mu}_{\boldsymbol{h}}^{(l)}(\boldsymbol{x}) + \boldsymbol{\tau}_{\boldsymbol{h}}^{(l)} \odot \boldsymbol{\sigma}_{\boldsymbol{h}}^{(l)}$ where $\boldsymbol{\tau}_{\boldsymbol{h}}^{(l)} = [\tau_{h_1}^{(l)}, \ldots, \tau_{h_p}^{(l)}]^\top \in \mathbb{R}_+^p$ is the related trust level. In this context, for each $i = 1, \ldots, p$, the approximated feasible domain is given by $\alpha_{h_i}^{(l)} = \tau_{h_i}^{(l)}\sigma_{h_i}^{(l)} - |\mu_{h_i}^{(l)}| \geq 0$, meaning that we allow to violate the approxi-

mated equality constraint $\mu_{h_i}^{(l)}$ up to the confidence level $\tau_{h_i}^{(l)}\sigma_{h_i}^{(l)}$. In other words, the equality constraints are approximated by a set of inequality constraints as follows:

$$\Omega_h^{(l)} \quad = \quad \left\{ \boldsymbol{x} \in \Omega \ ; \ \boldsymbol{\tau_h^{(l)}} \odot \boldsymbol{\sigma_h^{(l)}}(\boldsymbol{x}) - \left| \boldsymbol{\mu_h^{(l)}}(\boldsymbol{x}) \right| \geq 0 \right\}. \tag{4}$$

Remark 3.1. *It is possible to express* $\Omega_h^{(l)}$ *using only equality constraints. In fact, for a given* $\boldsymbol{x} \in \Omega$ *and an equality constraint* h_i*, one can set* $\alpha_{h_i}^{(l)}(\boldsymbol{x})$ *such that whenever* $\mu_{h_i}^{(l)}(\boldsymbol{x}) + \tau_{h_i}^{(l)}\sigma_{h_i}^{(l)}(\boldsymbol{x}) \leq 0$*,* $\alpha_{h_i}^{(l)}(\boldsymbol{x})$ *is set to* $\mu_{h_i}^{(l)}(\boldsymbol{x}) + \tau_{h_i}^{(l)}\sigma_{h_i}^{(l)}(\boldsymbol{x})$*, if* $\mu_{h_i}^{(l)}(\boldsymbol{x}) - \tau_{h_i}^{(l)}\sigma_{h_i}^{(l)}(\boldsymbol{x}) \geq 0$*, then* $\alpha_{h_i}^{(l)}(\boldsymbol{x}) = \mu_{h_i}^{(l)}(\boldsymbol{x}) - \tau_{h_i}^{(l)}\sigma_{h_i}^{(l)}(\boldsymbol{x})$*, and otherwise* $\alpha_{h_i}^{(l)}(\boldsymbol{x})$ *is set to* 0*. I.e.,*

$$\Omega_h^{(l)} = \left\{ \boldsymbol{x} \in \Omega \ ; \ \max \left[[\boldsymbol{\mu_h^{(l)}}(\boldsymbol{x}) + \boldsymbol{\tau_h^{(l)}} \odot \boldsymbol{\sigma_h^{(l)}}(\boldsymbol{x})]^-, \ \boldsymbol{\mu_h^{(l)}}(\boldsymbol{x}) - \boldsymbol{\tau_h^{(l)}} \odot \boldsymbol{\sigma_h^{(l)}}(\boldsymbol{x}) \right] = 0 \right\},$$

where we use $[s]^-$ *to denote the element-wise operation* $\min(0, s)$ *and the* max *for the element-wise maximum operator. In our preliminary tests, the obtained results with this choice of* $\Omega_h^{(l)}$ *turn to be less competitive compared to the use of the definition given by (4).*

Remark 3.2. *Using the UTB criterion, the obtained maximization sub-problem can be seen as a generalization of the original SEGO sub-problem formulation. In fact, by setting* $\boldsymbol{\tau_g^{(l)}}$ *and* $\boldsymbol{\tau_h^{(l)}}$ *to zero, one gets exactly the SEGO maximization sub-problem* (2)*.*

The use of the UTB feasibility criterion is meant to enlarge the regarded feasible domain during the first stages of the optimization process (where the size of the DoE is still small and the uncertainties are large). In what comes next, we will use SEGO-UTB to denote the SEGO framework when the UTB feasibility criterion is used in the constraints formulation of the maximization sub-problem. The full description of the SEGO-UTB framework is given in Algorithm 2.

Figure 2 illustrates the iterative process of SEGO and SEGO-UTB for three chosen iterations (first, second and final iterations) on the modified Branin problem. The feasible domain evolves with the GPs associated to the constraints

Algorithm 2 The SEGO-UTB framework.

input : Objective and constraints functions, initial DoEs for objective and constraints, a maximum number of iterations max_nb_it and an evolution strategy for the constraints learning rate.

1: **for** $l = 0$ **to** max_nb_it - 1 **do**

2: Build the surrogate models using GPs.

3: Set

$$\boldsymbol{x}^{(l+1)} = \arg \max_{\boldsymbol{x} \in \Omega_h^{(l)} \cap \Omega_g^{(l)}} \alpha_f^{(l)}(x),$$

where the expression of $\alpha_f^{(l)}$ is given in 5, $\Omega_g^{(l)}$ and $\Omega_h^{(l)}$ are given by (3) and (4), respectively.

4: Evaluate the objective and constraints functions at $\boldsymbol{x}^{(l+1)}$.

5: Update the DoE.

6: **end for**

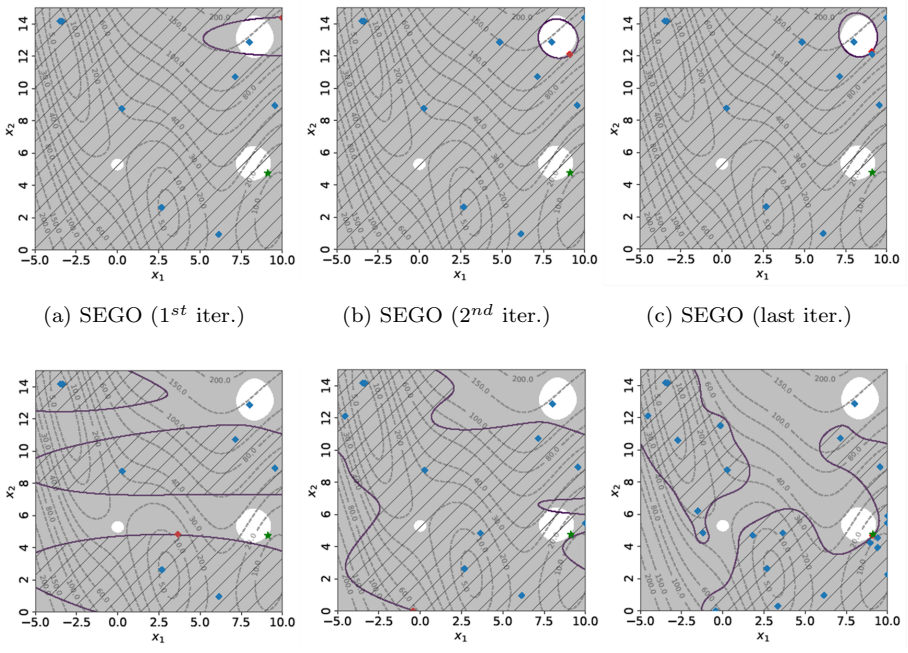
output : The best point found in the DoE

and the proposed learning rates. Clearly, by including the UTB, SEGO-UTB is able to explore more the feasible domain. It converges to the global minimum while SEGO could not explore the whole feasible domain. Therefore, although SEGO is showing a fast convergence (as it requires only 3 iterations), it reaches only a local minimum.

For a given iteration l, the update strategy of the constraints learning rates $\boldsymbol{\tau_g}^{(l)}$ and $\boldsymbol{\tau_h}^{(l)}$, turns to be an efficient tool to control the trade-off between exploration of the design space and the minimization of the objective function in the SEGO-UTB framework. In the next subsection, different strategies for updating $\boldsymbol{\tau_g}^{(l)}$ and $\boldsymbol{\tau_h}^{(l)}$ are introduced.

3.2. On the update of the constraints learning rate

For simplicity reasons, the overall constraints function is denoted, for a given $\boldsymbol{x} \in \Omega$, by $\boldsymbol{c}(\boldsymbol{x}) = [\boldsymbol{g}(\boldsymbol{x})^\top, \boldsymbol{h}(\boldsymbol{x})^\top]^\top \in \mathbb{R}^{m+p}$. For a given iteration l, let $\boldsymbol{\mu_c}^{(l)} : \mathbb{R}^d \to \mathbb{R}^{m+p}$ and $\boldsymbol{\sigma_c}^{(l)} : \mathbb{R}^d \to \mathbb{R}^{m+p}$ denote the mean and the standard deviation functions defining the GPs of the constraints $\boldsymbol{c}$, and let $\boldsymbol{\tau_c}^{(l)} =$



(a) SEGO (1^{st} iter.) (b) SEGO (2^{nd} iter.) (c) SEGO (last iter.)

(d) SEGO-UTB (1^{st} iter.) (e) SEGO-UTB (2^{nd} iter.) (f) SEGO-UTB (last iter.)

Figure 2: An illustration of three iterations of SEGO and SEGO-UTB (with a $\tau_g = 3$) on the modified Branin problem. The hatched area is the SEGO/SEGO-UTB unfeasible domain, the grey area shows the true unfeasible domain and the dashed curves are the contour plots of the objective function. The blue squares are the current DoE whereas the green star indicates the global minimum of the problem. The red square is the new point to add in the DoE.

$[[\boldsymbol{\tau_g}^{(l)}]^\top, [\boldsymbol{\tau_h}^{(l)}]^\top]^\top \in \mathbb{R}^{m+p}$ be the associate constraints learning rate as given by the UTB feasibility criterion.

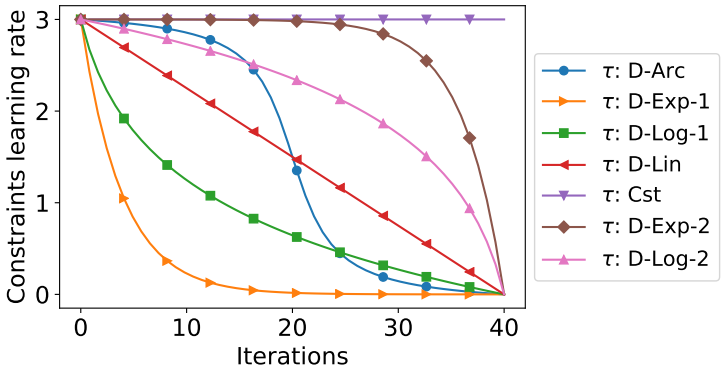
In general, the acquisition function α_f ensures that the exploration and exploitation trade-off is respected during the enrichment procedure. Hence one can assume that the sample points are somehow well distributed in the design space during the optimization process. In this context, one can use different strategies for updating the constraints learning rate $\tau_{c_i}^{(l)}$ (associated to the constraint c_i). A first trivial strategy, is ensured by making the value of $\tau_{c_i}^{(l)}$ constant all over the iterations. This choice is motivated by the fact that $\tau_{c_i}^{(l)}$ is used to scale $\sigma_{c_i}^{(l)}$, and the latter function decreases systematically whenever

the model is getting accurate. This updating strategy is noted (Cst.). In this case, a natural constant choice for $\tau_{c_i}^{(l)}$ is 3 for all $i = 1, \ldots, m+p$ and iteration index l. With this value, the trust interval over all the reliable GPs is expressed with 99% trust for a single point $\boldsymbol{x}$ sampled in Ω. We note that, in case of only inequality constrained optimization problems, working with a fixed $\tau_{c_i}^{(l)}$ was also proposed in [16, 18].

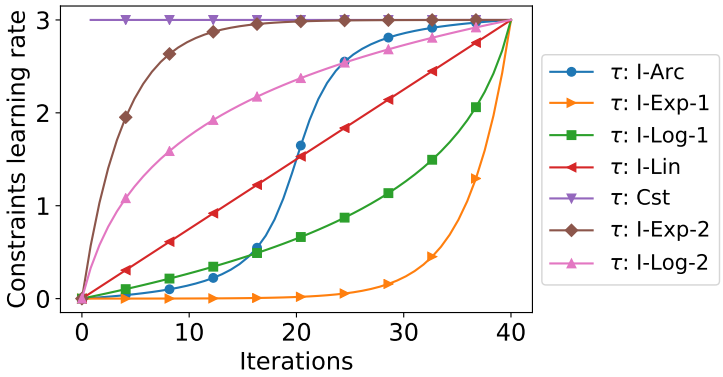
A second possible strategy can be as follows. In fact, as the quality of the GP approximation will most likely depend on the size of DoE, we expect that the larger is the size of the DoE, the better is the GP approximation. This suggests naturally to be more confident on the GP prediction when the number of sample points in the DoE increases. Hence, for a given $i = 1, , \ldots, m+p$, the constraints learning rate $\tau_{c_i}^{(l)}$ (associated to the constraint c_i) should be reduced as far as the number of points of the DoE increases (i.e., l getting larger). Assuming that the maximum number of iterations (see max_nb_it in Algorithm 1) is large enough, so that the GP approximations can be considered very accurate, the constraints learning rate can be decreased systematically (from a given initial value, typically $\tau_{c_i}^{(0)} = 3$) to reach zero at the end of the optimization process (i.e., $\tau_{c_i}^{(\text{max_nb_it})} = 0$).

Figure 3a illustrates different decreasing monotonic profiles for each component of the constraints learning rate: arc-tangent (Arc), linear (Lin), two logarithmic (Log), and two exponential (Exp) profiles for $\tau_{c_i}^{(l)}$. Within such updating strategies, for each component, the constraints learning rate is reduced regardless the quality of the GP approximation of the constraint during the optimization process.

In our numerical tests, we noticed that SEGO performs well during the early stages of the optimization; this remark motivates the following updating strategy of the constraints learning rate. In fact, we will try to mimic SEGO during the first iterations and, then, incorporate gradually the uncertainties on the GPs of the constraints in order to encourage a better exploration of the feasible domain. We obtain, for each constraint, a non-decreasing learning rate with respect to the outer iterations of the regarded algorithm. Namely, at the first iteration,



(a) Decreasing strategy



(b) Non-decreasing strategy

Figure 3: Evolution of the constraints learning rate over iterations (with max_nb_it = 40).

the constraints learning rate related to each constraint is first set to zero and then systematically increased to reach a maximum value (typically, at the end of the optimization, one would have $\tau_{c_i}^{(\text{max_nb_it})} = 3$). Different non-decreasing strategies can be used; see Figure 3b for the non-decreasing trends tested in the context of this paper.

In the next section, different decreasing and non-decreasing strategies for the constraints learning rate are tested on well-know mixed-constrained test problems. A comparison with the state-of-the-art solvers is also included.

4. Numerical tests

In this section, the potential of the proposed algorithm is evaluated using an extensive test set of 29 problems.

4.1. Implementation details

Our implementation choices for SEGO-UTB and SEGO are as follows. For the GPs, we choose to work with linear regression trend and a Gaussian correlation function, all based on the open-source Python *surrogate modeling toolbox* (SMT) [24]. The choice of all the hyper-parameters is handled by the default settings of the toolbox. As acquisition function, we used the explicit *scaled Watson Barnes* (WB2S) [21] . For a given iteration l, the WB2S infill criterion is given by

$$\alpha_f^{(l)}(\boldsymbol{x}) = s^{(l)}\ \mathrm{EI}^{(l)}(\boldsymbol{x}) - \mu_f^{(l)}(\boldsymbol{x}), \tag{5}$$

where $\mathrm{EI}^{(l)}(\boldsymbol{x})$ is the expected improvement function [14] at $\boldsymbol{x}$ and $\mu_f^{(l)}(\boldsymbol{x})$ is the mean function of the GP model of f at the iteration l. To define the scale factor $s^{(l)}$, we first compute $\boldsymbol{x}_{\mathrm{EI}_{\max}} = \arg\max_{\boldsymbol{x}} \mathrm{EI}(\boldsymbol{x})$, then we set $s^{(l)}$ to $100\frac{\left|\mu_f^{(l)}(\boldsymbol{x}_{\mathrm{EI}_{\max}})\right|}{\mathrm{EI}(\boldsymbol{x}_{\mathrm{EI}_{\max}})}$ if $\mathrm{EI}\left(\boldsymbol{x}_{\mathrm{EI}_{\max}}\right) \neq 0$, and to 1 otherwise. In our case, the point $\boldsymbol{x}_{\mathrm{EI}_{\max}}$ is chosen as the point that maximizes the EI function among $100d$ points from the design space (generated using Latin Hyper-cube Sampling strategy).

During the optimization process, the points in the DOE are not necessary feasible. In fact, the infill criterion WB2S can be evaluated even if all the points in the DOE are infeasible [21]. However, at the end of the optimization, if all the points in the DOE are infeasible, SEGO-UTB and SEGO will return the point with the minimal constraints violation. The initial (resp. final) value associated with the decreasing (resp. non-decreasing) strategies for all components of the constraint learning rates τ_c is set to 3. The optimization sub-problem (2) is solved in two steps. The first consists in finding a warm starting point by solving (2) with the *Improved Stochastic Ranking Evolution Strategy* (ISRES) [25] of the Python NLopt package [26]. The solution point is then obtained by solving (2) using SNOPT [27], from the PyOptSparse toolbox [28], where the starting point is set as the solution returned by ISRES.

4.2. Solvers in the comparison

In this paper, SEGO-UTB is compared to some of the state-of-the-art BO solvers (in addition to SEGO):

- ALBO: a BO solver using an augmented Lagrangian approach [12],
- SUR: a BO solver using a stepwise uncertainty reduction [29],
- PESC: a BO solver based on the predictive entropy search [30],
- EFI: a BO solver based on the expected feasibility improvement [31].

The solvers ALBO, EFI and SUR were taken from the DiceOptim R package [32] while the PESC solver was taken from the Spearmint Python toolbox[1]. All the parameters of those BO solvers were kept unchanged except the the correlation function which is set to be Gaussian. In fact, the proposed default choice for the correlation function (a Matérn correlation function) did not perform well in our numerical tests.

For completeness, two well-known derivative free solvers are also included in the comparison:

- NOMAD: a mesh adaptive direct search solver [33, 34]. The default parameters are kept unchanged.
- COBYLA: a trust-region solver based on linear approximations [35]. We worked with the Scipy Python toolbox implementation of COBYLA [36].

We note that SUR, PESC, EFI, NOMAD and COBYLA, handle only inequality constraints. To manage equality constraints, each constraint of the form $\boldsymbol{h}(\boldsymbol{x}) = 0$ are changed into two inequality constraints of the form $\boldsymbol{h}(\boldsymbol{x}) \geq 0$ and $\boldsymbol{h}(\boldsymbol{x}) \leq 0$.

We stress also that, unlike all the tested BO solvers where one starts with an initial DoE, NOMAD and COBYLA require only a single initial point to start

[1] https://github.com/HIPS/Spearmint

the optimization procedure. To not penalize the latter two solvers, we choose the best valid point in the initial DoE as a first guess for NOMAD and COBYLA. If there is no valid point, the best point (i.e., with the smallest violation of the constraints) in the initial DoE is chosen.

Last, for all the tested solvers including SEGO and SEGO-UTB, two tolerances on the violation for each of constraints are considered, namely, $\epsilon_c = 10^{-2}$ and $\epsilon_c = 10^{-4}$.

4.3. Comparison results using convergence plots

In this subsection, we will analyse the performance of all the tested solvers using convergence plots related to a set of four known problems.

4.3.1. Test problems

Among the four tested problems, three are taken from Picheny et al. [12] for which ALBO is in particular very competitive. One of the problems, is the *Linear-Hartman-Ackley* (LAH) test case, which has four design variables, a linear objective function, one equality constraint (given by the Hartman function) and one inequality constraint (given by the Ackley function). The second mixed optimization problem has two design variables with a re-scaled version of the "Goldstein-Price" function as objective function. The problem is constrained with one inequality constraint (given by a sinusoidal function) and two equality constraints (using a centered "Branin" function and a function taken from Parr et al. [37]); henceforth, this problem is named GBSP. The third problem involved two design variables, a linear objective function, a sinusoidal and a quadratic inequality constraints; henceforth named LSQ. The additional fourth problem is the *Modified Branin* (MB) test case [22]. This problem has two design variables, a non-linear objective function and one inequality constraint. The full expressions of these problems can be found in Appendix A.1.

4.3.2. Convergence plots

Similarly to Picheny et al. [12], we will build four convergence plots by problem to assess the good performance of our solver. For each problem, the convergence

plots are built on the following way. First, we perform 100 independent runs for each solver using 100 different initial DoEs using the Latin Hypercube Sampling method. For each run, all the solvers are initiated with the same DoE. The size of the initial DoEs and the maximum number of iterations are respectively set to $n_{start} = \max(d+1, 5)$ and $max_nb_it = 40d - n_{start}$ where d is the dimension of the regarded problem (meaning a total budget of $40d$ function evaluations).

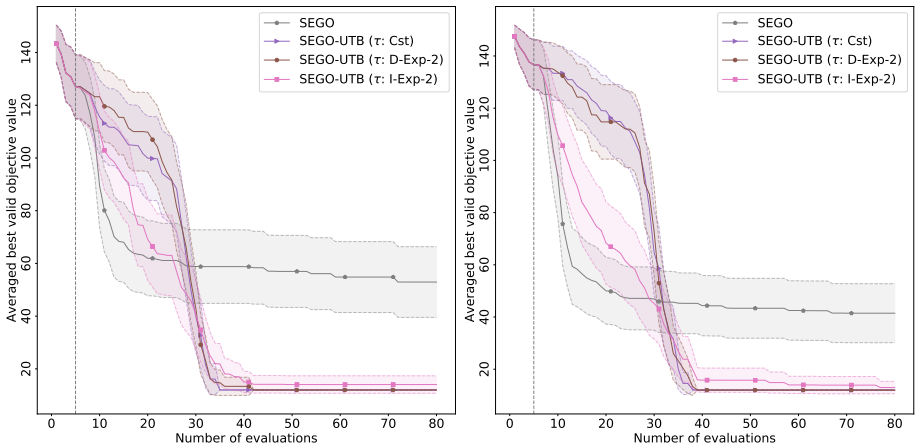
All the solvers are then compared by displaying the average and the standard deviation, up to a scaling factor, of the best values over the 100 runs for increasing number of evaluations. The best value is defined as the best valid value if there is, at least, one valid point in the DoE, otherwise, a penalization replaces the obtained invalid value. The penalization is set to 3 for LAH and GBSP, 2 for LSQ and 150 for the MB problem.

4.3.3. Results

In what follows, we stress that, concerning SEGO-UTB, different evolution strategies for updating the constraints learning rate are tested (as given in Figure 3). For clarity reasons, only the best compromises among the non-decreasing and decreasing strategies are considered meaning SEGO-UTB (τ: D-Exp-2) and SEGO-UTB (τ: I-Exp-2). The constant constraints learning rate evolution (which corresponds to the strategy proposed in [16] when only inequality constraints are present) is also included in the comparison. For completeness, the obtained results using all the constraints learning rate strategies can be found in Appendix B.

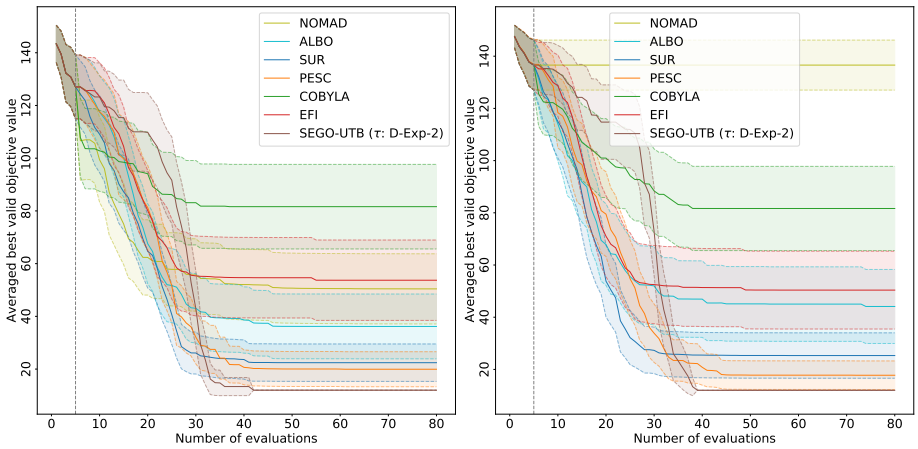
The obtained results considering two tolerances on the violation of the constraints are presented on the following way. First, we confront the SEGO-like solvers (namely, SEGO, SEGO-UTB (τ: Cst), SEGO-UTB (τ: D-Exp-2) and SEGO-UTB (τ: I-Exp-2)). Then, the best SEGO-like solver is compared to the other solvers.

Figure 4 depicts a comparison between SEGO and SEGO-UTB on the MB optimization problem. Clearly, all the three SEGO-UTB variants are outperforming SEGO for both levels of accuracy on the constraints violation. In



(a) $\epsilon_c = 10^{-2}$. (b) $\epsilon_c = 10^{-4}$.

Figure 4: Convergence plots for the MB problem on the SEGO-like solvers, considering the two levels of constraints violation 10^{-4} and 10^{-2}. The vertical grey-dashed line outlines the number of points in the initial DoEs.



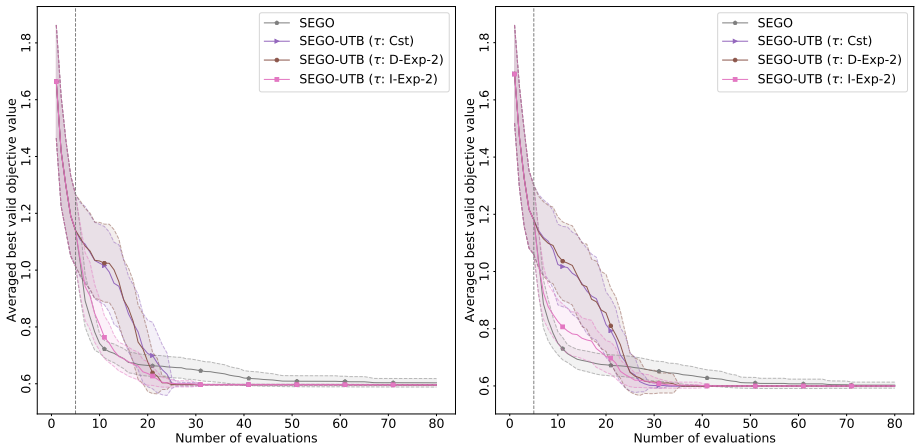
(a) $\epsilon_c = 10^{-2}$. (b) $\epsilon_c = 10^{-4}$.

Figure 5: Convergence plots for the MB problem NOMAD, COBYLA and BO solvers, considering the two levels of constraints violation 10^{-4} and 10^{-2}. The vertical grey-dashed line outlines the number of points in the initial DoEs.

fact, the averaged best valid value of SEGO does not converge to the same value as SEGO-UTB. Note also that SEGO displays a high standard deviation which means that it is not targeting all the time the same solution. On the contrary, the SEGO-UTB variants are all targeting the global minimum zone. Including uncertainties in the constraints models is leading to a better exploration of the feasible domain. Also, one can see that SEGO-UTB (τ: D-Exp-2) displays the best performance. For that reason, Figure 5 shows a comparison of SEGO-UTB (τ: D-Exp-2) and the other tested solvers using the two considered constraints violations. Clearly, NOMAD and COBYLA are performing the worst among all the tested solvers, in particular, when a strict tolerance on the constraints violation is used (see Figure 5b). The high associated standard deviation indicates that some of the runs converged to different values. The other tested BO solvers show a better performance although PESC, ALBO, SUR and EFI did not reach the global minimum. In terms of the convergence speed, SEGO-UTB (τ: D-Exp-2) is being the fastest independently of the constraints violation tolerance.

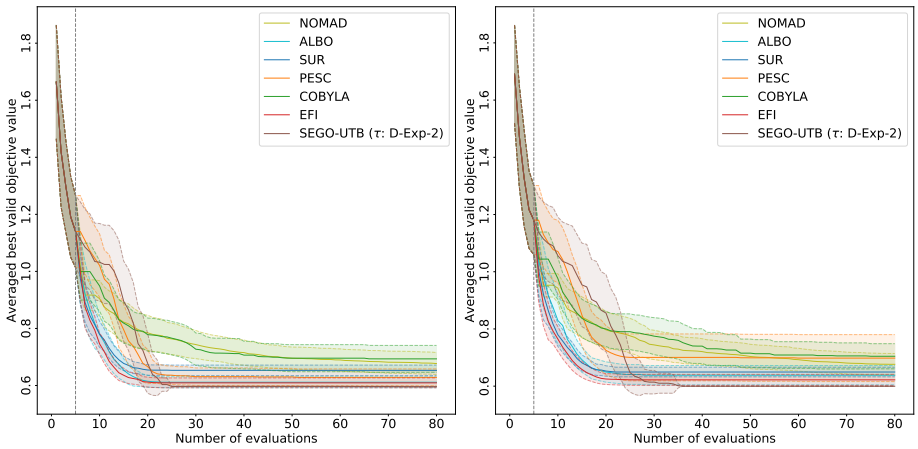
In Figures 6 and 7, we present the obtained results on the LSQ problem. One can see that all the SEGO-like solvers are converging to the global minimum of this problem (see Figure 6). SEGO-UTB (τ: I-Exp-2) is exhibiting a slightly better performance in term of convergence speed. In the comparison with the other solvers, see Figure 7, COBYLA and NOMAD are displaying the worst performance. Figure 7 shows that PESC has better convergence properties with a high tolerance on the constraints violation. Similarly to SEGO-UTB (τ: D-Exp-2), the solvers ALBO, SUR and EFI are all converging to the global minimum in average except some runs; since their standard deviation is not converging to zero at the end of the optimization.

For the GBSP problem (which has mixed constraints), the obtained results are given in Figures 8 and 9. SEGO displays good performance during the early stages of the optimization but does not reach the global minimum of the GBSP problem for both tolerances on the constraints violations. The SEGO-UTB variants are able to explore better the feasible domain (with a slower convergence rate compared to SEGO) and converge to the global minimum for all runs



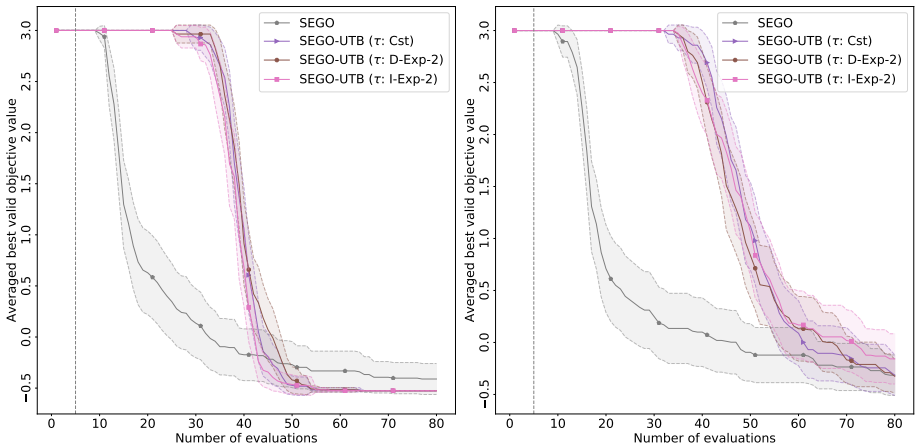
(a) $\epsilon_c = 10^{-2}$. (b) $\epsilon_c = 10^{-4}$.

Figure 6: Convergence plots for the LSQ problem on the SEGO-like solvers, considering the two levels of constraints violation 10^{-4} and 10^{-2}. The vertical grey-dashed line outlines the number of points in the initial DoEs.



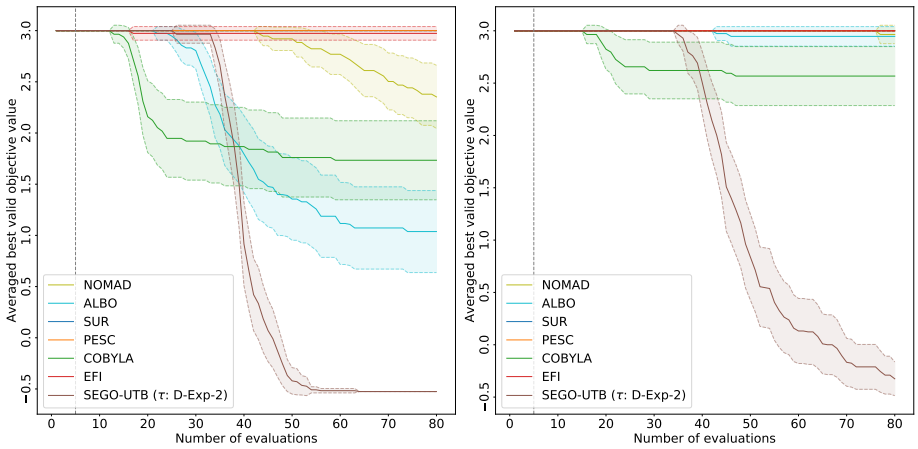
(a) $\epsilon_c = 10^{-2}$. (b) $\epsilon_c = 10^{-4}$.

Figure 7: Convergence plots for the LSQ problem NOMAD, COBYLA and BO solvers, considering the two levels of constraints violation 10^{-4} and 10^{-2}. The vertical grey-dashed line outlines the number of points in the initial DoEs.



(a) $\epsilon_c = 10^{-2}$. (b) $\epsilon_c = 10^{-4}$.

Figure 8: Convergence plots for the GBSP problem on the SEGO-like solvers, considering the two levels of constraints violation 10^{-4} and 10^{-2}. The vertical grey-dashed line outlines the number of points in the initial DoEs.



(a) $\epsilon_c = 10^{-2}$. (b) $\epsilon_c = 10^{-4}$.

Figure 9: Convergence plots for the GBSP problem NOMAD, COBYLA and BO solvers, considering the two levels of constraints violation 10^{-4} and 10^{-2}. The vertical grey-dashed line outlines the number of points in the initial DoEs.

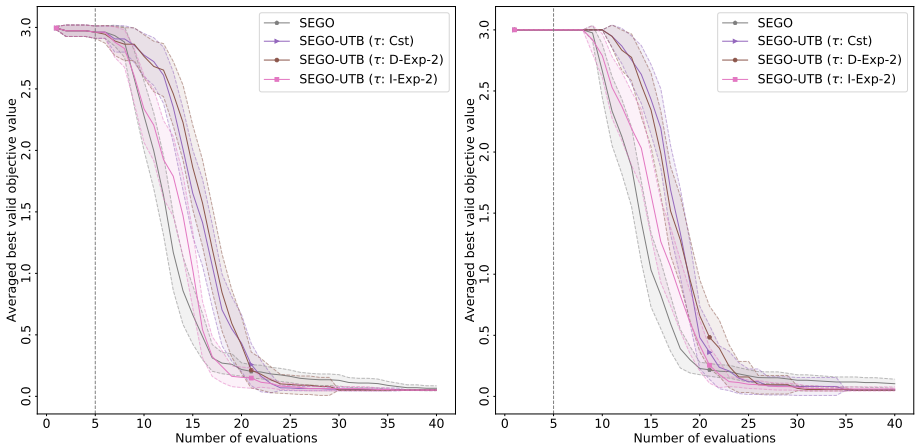
(see Figure 8a) considering the constraints violation of 10^{-2}. Figure 8b shows that the use of a stricter tolerance on the constraints violation deteriorates the convergence of the SEGO-UTB solvers although they still converge close to the SEGO solution. In comparison with the other solvers, see Figure 9, the SEGO-like solvers are outperforming (by far) all the tested solvers. We note also that the performances of the solvers ALBO, NOMAD and COBYLA turn to be very sensitive to the regarded value of the constraints violation tolerance. In fact, ALBO, NOMAD and COBYLA display better performance when using a large tolerance, such performances get worst when the tolerance on the violation of the constraints is stricter. Overall, the obtained results, in particular, confirm the efficiency of SEGO-UTB when handling problems with both equality and inequality constraints. Again, see Figure 11, the superior performance of SEGO-UTB is confirmed on the LAH problem (which is mixed constrained) with the two levels of constraints violations.

In this subsection, the obtained results showed the potential of including the uncertainty while modeling the constraints during the optimization procedure. In particular, one was able, using convergence plots, to show the good performance of the proposed method compare to the state-of-the-art BO solvers on four known test problems. In the next subsection, we will use data profiles (adapted to our constrained setting) to confirm the efficiency of the proposed method, all using a larger test pool formed by 29 test problems.

4.4. Comparison results using data profiles

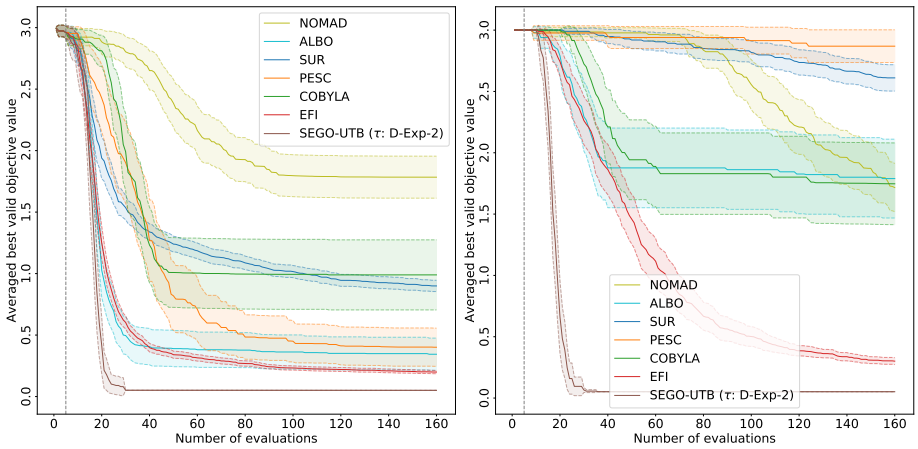
4.4.1. Problem instances

Our benchmark set is composed of 29 optimization problems, from [12, 22, 38, 39], which are mixed constrained (up to 38 equality and inequality constraints) of 2 to 10 design variables. A detailed description of the test problems is given in Appendix A.2. Our test set can be divided into two classes. The first one, referred as the *weakly non-linear constrained* (WNLC) set, is composed of 16 constrained optimization problems where the constraints are linear or quadratic. The second class is composed of the rest of the problems in our test pool, this



(a) $\epsilon_c = 10^{-2}$. (b) $\epsilon_c = 10^{-4}$.

Figure 10: Convergence plots for the LAH problem on the SEGO-like solvers, considering the two levels of constraints violation 10^{-4} and 10^{-2}. The vertical grey-dashed line outlines the number of points in the initial DoEs.



(a) $\epsilon_c = 10^{-2}$. (b) $\epsilon_c = 10^{-4}$.

Figure 11: Convergence plots for the LAH problem NOMAD, COBYLA and BO solvers, considering the two levels of constraints violation 10^{-4} and 10^{-2}. The vertical grey-dashed line outlines the number of points in the initial DoEs.

class is referred to the *highly non-linear constrained* (HNLC) problems.

Due to the stochastic nature of the BO solvers, we create instances for the tested problems. In fact, the obtained results for a given problem may depend on the choice of the chosen initial DoEs. Thus, an instance of the problem is related to the choice of an initial DoE. In our case, we generate 10 different initial DoEs for each problem which leads to the creation of 290 problem instances.

4.4.2. Data profiles

Data profiles [40] are designed for derivative-free optimization, to show how well a solver performs, given some computational budget, when asked to reach a specific reduction in the objective function value, measured in our case by

$$\tilde{f}(x^{(l)}) - f_{\text{opt}} \leq \epsilon(|f_{\text{opt}}| + 1), \tag{6}$$

where $\epsilon \in [0, 1]$ is the required level of accuracy, f_{opt} represents the best objective value found (within the feasible domain) by all solvers tested for a specific problem and within a given maximal computational budget. $\tilde{f}(x^{(l)})$ is set to the value of the objective function at the iteration l if $x^{(l)}$ satisfies the constraints, and set to $+\infty$ otherwise. We note that we had to adapt the data profiles [40] to our constrained setting; for that reason, we are using the same convergence test as proposed in [41].

Data profiles plot the percentage of problems solved by the solvers under consideration for different values of the computational budget. Let $\mathcal{S}$ be the set of the tested solvers and $\mathcal{P}$ the set of the problem instances. A data profile is computed, for each solver $s \in \mathcal{S}$, as the percentage of the problem instances that can be solved within κ objective function evaluations:

$$\frac{1}{|\mathcal{P}|} \text{ size} \left\{ p \in \mathcal{P} \quad : \quad \frac{t_{p,s}}{d_p} \leq \kappa \right\},$$

where d_p is the dimension of the problem instance $p \in \mathcal{P}$, $t_{p,s}$ is the number of function evaluations required by solver $s \in \mathcal{S}$ on problem instance p to satisfy the convergence test (6) for a given tolerance ϵ. If the convergence test is not satisfied after the maximum budget of function evaluations, $t_{p,s}$ is set to $+\infty$.

The units budget are expressed with d_p to allow the combination of problems of different dimensions in the same profile.

We used in our experiments a maximal computational budget consisting of $40d_p$ function evaluations, as we are primarily interested in the behavior of the algorithms for problems where the evaluation of the objective function is expensive. For the level of accuracy used in the convergence test (6), we set $\epsilon = 10^{-3}$.

4.4.3. Results

We note that, due to the dimension of the tested problems and the number of constraints, all the tested BO optimizers (except the SEGO-like solvers) did not give good results and were computationally very expensive. Table 1 shows the CPU-time average using 10 runs for the BO solvers on five problems using a budget of $40d$ maximum function evaluations. Clearly, one can see that the solvers ALBO, SUR, PESC and EFI run on the G07 problem are very consuming in terms of CPU-time. Thus, those solvers cannot be tested on the all 29 problems in a reasonable time. We note also that the solver SUR cannot handle more than four constraints (which is the case of many problems in our test bed) in the DiceOptim implementation. For all these reasons, we consider that the BO solvers, ALBO, SUR, PESC and EFI are not adapted for our test problems, the presented results include only the solvers: NOMAD, COBYLA, SEGO and SEGO-UTB.

For clarity reasons (similarly to Section 4.3), we test different evolution strategies for the constraints learning rate within the SEGO-UTB solver, but only the best among the non-decreasing and decreasing strategies are kept. The constant constraint learning rate evolution is also included in our tests. The complete results for decreasing and non-decreasing strategies are given in Appendix B. Note that all these tests are performed for the two levels of constraints violations 10^{-2} and 10^{-4}.

We now comment on the data profiles obtained by the regarded solvers. Figure 12a depicts the obtained data profiles when considering all the tested

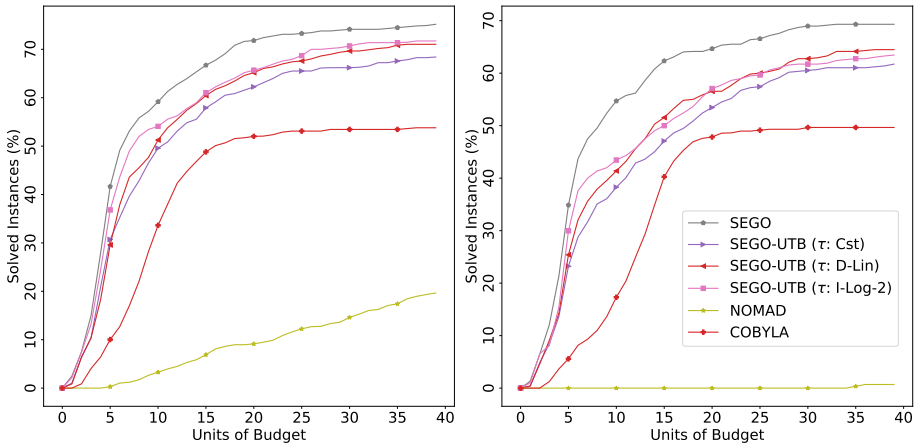
Table 1: CPU-time average (in seconds) using 10 runs for the BO solvers on five problems using a budget of $40d$ maximum function evaluations.

Problem	SEGO	SEGO-UTB	ALBO	SUR	EFI	PESC
LAH	203.64	2 708.99	5 131.97	2 592.32	1 212.04	2 729.64
GBSP	164.84	266.76	608.33	1 510.61	497.04	777.06
LSQ	157.43	177.04	833.83	417.97	207.82	592.70
MB	79.10	93.52	328.41	1 059.24	535.18	615.98
G07	2 098.86	4 245.55	611 025.72	–	1 195 346.89	55 755.87

problems considering the constraints violation $\epsilon_c = 10^{-2}$. As in the convergence plots tests, the SEGO-like solvers appear as the best. In fact, using a maximal budget, the SEGO-like solvers are able to solve around 70% the tested instances, COBYLA solves 53% and NOMAD around 20%. For smaller units of budget, the gap between the SEGO-like solvers is similar. Note that, for large budgets, SEGO displays slightly better performance compared to the other SEGO-UTB solvers. Typically, at the end of the optimization, SEGO is able to solve 75% of the instances while SEGO-UTB (τ : I-Log-2) is solving 72%. The SEGO-like solvers outperform COBYLA and NOMAD even when a stricter tolerance on the constraints violation is used, see Figure 12b. For instance, with the maximal budget and using a tolerance $\epsilon_c = 10^{-4}$ on the violation of the constraints, SEGO is outperforming all the tested solvers by solving 70% of the tested instances. In all our tests, the use of stricter tolerance on the constraints violation reduced the percentage of the instances solved by each solver. We acknowledge also that NOMAD is not well adapted for equality constraints, this, in particular, explains the bad performance of NOMAD when a stricter tolerance on the violation of the constraints is used.

For a deeper analysis, we plot two data profiles respectively using the two problem classes WNLC and HNLC The obtained profiles are depicted in Figures 13 and 14. Clearly, one can see that SEGO is solving more instances than SEGO-UTB on WNLC problems with both constraints violations. On the contrary,

SEGO-UTB solvers are showing similar slightly better performances on the HNLC problems.



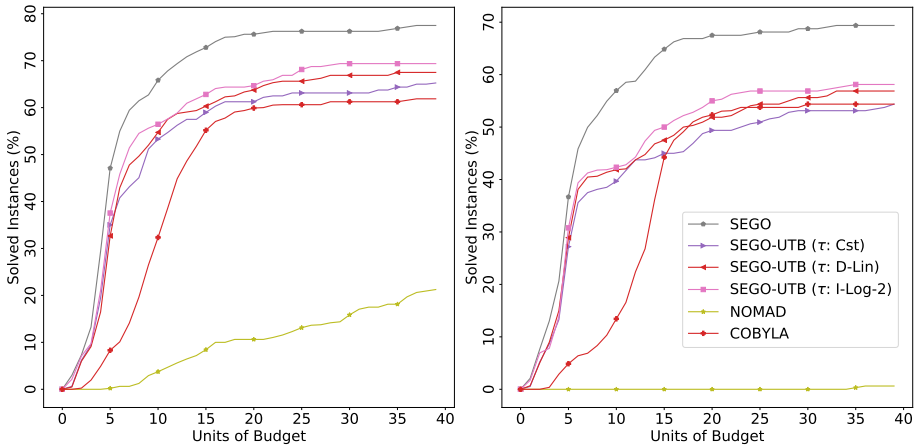
(a) $\epsilon_c = 10^{-2}$. (b) $\epsilon_c = 10^{-4}$.

Figure 12: Obtained data profiles of the 29 problems, considering the two levels of constraints violation 10^{-4} and 10^{-2}.

Overall, one concludes that including the uncertainties of the constraints within the SEGO framework turns to offer a better exploration of the feasible domain, in particular when the constraints present high non-linearities.

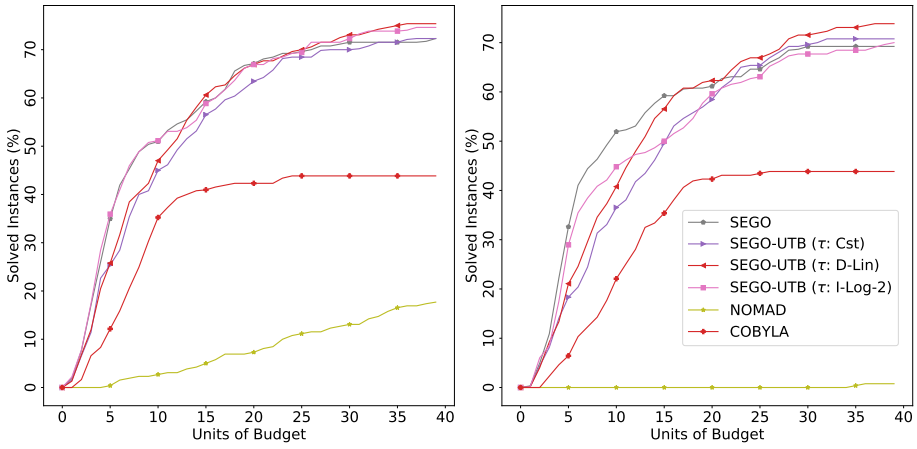
5. An application to Aircraft Design

This section presents an aircraft design application where the SEGO solvers lead to a significant improvement. We target to design a hybrid aircraft, featuring distributed electric propulsion [19, 42, 43], the related concept of such aircraft is shown in Figure 15. Its main feature is the propulsive chain, which is made up of turbo-generators and batteries, that supply electric power to the set of distributed ducted fans, placed along the wing. The two gas turbines are evident at the rear of the aircraft. Meanwhile, batteries are not shown since they are placed within the cargo bay. To reduce emissions, the aircraft is able to fly at



(a) $\epsilon_c = 10^{-2}$. (b) $\epsilon_c = 10^{-4}$.

Figure 13: Obtained data profiles of the WNLC problems, considering the two levels of constraints violation 10^{-4} and 10^{-2}.



(a) $\epsilon_c = 10^{-2}$. (b) $\epsilon_c = 10^{-4}$.

Figure 14: Obtained data profiles of the HNLC problems, considering the two levels of constraints violation 10^{-4} and 10^{-2}.

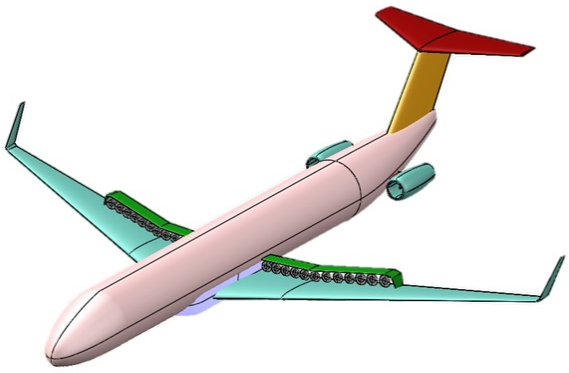


Figure 15: A Hybrid aircraft concept, featuring distributed propulsion [19].

least to 3000 ft in fully electric mode and it is designed to carry 150 passengers for a range of 1200 nmi.

The *fixed-wing aircraft sizing tool* (FAST) [44] is used to explore this aircraft concept and is fully coded in Python. It is based on engineering methods, to have reliable results with low computational cost [45]. Some modules have been modified to consider the features introduced by the hybrid chain [19, 43]. We note also that the unconventionality of the concept, showing more interaction between the disciplines, (i.e. for the aerodynamics and the propulsion), makes the MDO a powerful tool to explore their sizing [46, 47]. Thus it is a relevant case study for the BO solvers already introduced in this paper.

5.1. The aircraft optimization problem

The optimization problem consists in minimizing the total energy consumption (e.g. sum of the fuel and batteries energy) with respect to the geometry (e.g. geometrical parameters defining wing, horizontal and vertical tail). This objective will be denoted by TEC.

In particular, the wing is defined by surface S_w, aspect ratio AR_w, wing position x_w and sweep angle, computed at 25% of the chord, $\Lambda_{25,w}$. The horizontal and vertical tail are defined by their surfaces S_{HT} and S_{VT}, aspect ratios AR_{HT} and AR_{VT} and sweep angles $\Lambda_{25,HT}$ and $\Lambda_{25,VT}$, computed at

25% of the chord. To consider the propulsive aspects, battery volume τ_b is added as design variable too. Finally, the cruise altitude h_{cruise} belongs to design variables vector, to ensure that the aircraft flies at the maximum aerodynamic efficiency. With this notation, it is possible to write the design vector as $x = [x_w, S_w, AR_w, \Lambda_{25,w}, S_{HT}, AR_{HT}, \Lambda_{25,HT}, S_{VT}, AR_{VT}, \Lambda_{25,VT}, \tau_b, h_{cruise}]^\top$ that contains sub-vectors of the geometrical parameters for the wing, the horizontal and vertical tails.

The design space exploration is reported in Table 2 where the values are taken within common data for the type of aircraft considered [45].

Table 2: Design optimization space definition.

Variable		**Min.**	**Max.**	**Variable**		**Min.**	**Max.**
h_{cruise}	[kft]	27	35	S_{HT}	[m^2]	20	80
τ_b	[m^3]	1.5	3.0	AR_{HT}		3	5
x_w	[m]	15	18	$\Lambda_{25,HT}$	[°]	25	45
S_w	[m^2]	100	130	S_{VT}	[m^2]	20	50
AR_w		9	12	AR_{VT}		1	2
$\Lambda_{25,w}$	[°]	20	45	$\Lambda_{25,VT}$	[°]	30	45

The problem is subject also to constraints. To ensure the feasibility of the aircraft design: the wing has to store enough fuel for the whole mission ($MFW \geq m_f$, with MFW maximum fuel weight and m_f the mission fuel) and generate enough lift in approach condition for landing ($C_{L_{\max}} \geq C_{L_{app}}$, with $C_{L_{\max}}$ maximum and $C_{L_{app}}$ approach lift coefficient). Horizontal tail is designed to ensure takeoff rotation, meaning to have positive pitching moment for every center of gravity position [1] ($\mathcal{M}_{takeoff} = 0$). Vertical tail is instead designed to counterbalance the fuselage yaw moment in cruise [1] ($\mathcal{N}_{cruise} = 0$). The batteries are subject to two constraints, related to power and energy requirements. The first demands that they produce enough power at takeoff ($P_b \geq P_{takeoff}$), and the second ensures that, at the end of the flight, there is still a 20% energy available.

The parameter that controls the energy consumption is the state of charge SoC, defined as the ratio between the energy consumed and the total energy stored. With the SoC definition, the second condition can be written as $\text{SoC}_{fin} \geq 0.20$. The SoC_{min} limit is the safety margin for most of batteries, to not damage the system [48]. Another constraint related to stability is the static margin SM, which has to be included between $SM_{min} = 0.05$ and $SM_{max} = 0.10$, according to certification [1].

Regarding the restrictions coming from airport configuration, for the type of aircraft considered, the takeoff field length TOFL must not exceed $\text{TOFL}_{max} =$ 2.2 km and the wing span b_w is below $b_{w_{max}} = 36$ m [49]. Finally, the last constraint is given to ensure that the cruise altitude is chosen to maximize the efficiency, that is $C_{L_{cruise}} = C_{L_{opt}}$, where the left side is the cruise lift coefficient and the right side is the lift coefficient at which the maximum aerodynamics efficiency occurs. To sum up, for the regarded aircraft design problem, the equality constraints are represented by $\boldsymbol{h} = [\mathcal{M}, \mathcal{N}, C_{L_{cruise}} - C_{L_{opt}}]^\top$ while the inequality constraints are depicted as follows: $\boldsymbol{g} = [b_{w_{max}} - b_w, MFW - m_f, C_{L_{\max}} - C_{L_{app}}, \text{SoC} - \text{SoC}_{min}, P_b - P_{takeoff}, \text{TOFL}_{max} - \text{TOFL}, SM - SM_{min},\ SM_{max} - SM]^\top$. We thus obtain the following optimization problem:

$$\min \left\{ \text{TEC}(\boldsymbol{x}) \ \text{ w.r.t. } \ \boldsymbol{x} \in \mathbb{R}^{12} \ \text{ s.t. } \ 0 \leq \boldsymbol{g}(\boldsymbol{x}) \in \mathbb{R}^8 \text{ and } 0 = \boldsymbol{h}(\boldsymbol{x}) \in \mathbb{R}^3 \right\}$$

We note that in the numerical tests presented in Section 4.3 we tolerate a violation of the constraints up to 10^{-2}. However, in the regarded aircraft optimization design problem, the tolerated violation on the constraints $\boldsymbol{h}$ and $\boldsymbol{g}$ is driven by the physical properties of the problem [45], respectively, as follows $\epsilon_h = [10^{-2}, 10^{-2},\ 5000]^\top$ and $\epsilon_g = [10^{-2}, 100, 10^{-2}, 10^{-2}, 1000, 100, 10^{-2}, 10^{-2}]^\top$. If a point, during the optimization process, exceeds the tolerated constraints violation, the objective function is penalized with the value 4.10^5. All the other implementation choices are kept as explained in Section 4.4.2 for all the tested solvers. The presented convergence plot for each solver is built using 10 runs.

5.2. Results

Similarly to the reported results in Section 4.4.2, only the SEGO-like solvers, COBYLA and NOMAD will be included in the presented comparison. The other BO solvers, due to the dimension of the regarded problem, were very consuming in CPU and did not lead to any acceptable results in a reasonable time.

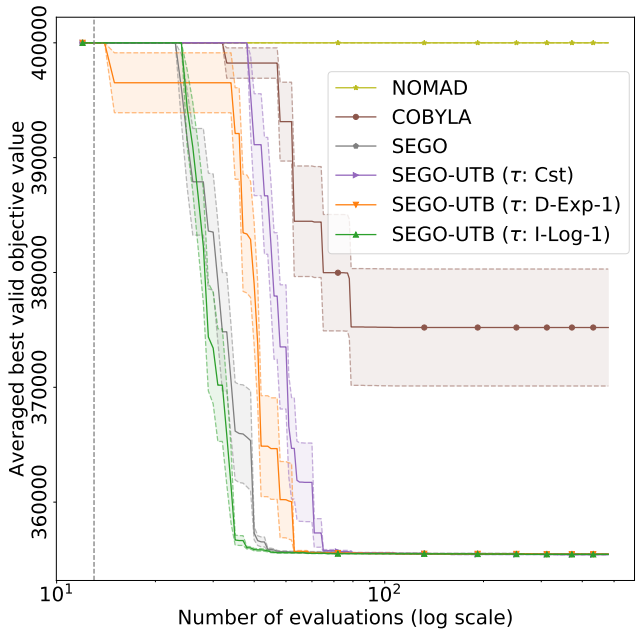


Figure 16: Convergence plots for the FAST problem. The vertical grey-dashed line outlines the number of points in the initial DoEs.

Figure 16 depicts the obtained convergence plots while solving the FAST optimization problem. One can see that the SEGO-like solvers are outperforming COBYLA and NOMAD. In fact, NOMAD was never able to find any feasible point for the 10 optimization runs performed. The high standard deviation of COBYLA implies that it does not converge to the same optimum for each run. Among the SEGO-like solvers, SEGO-UTB (τ: I-Log-1) appears to converge the fastest to the best solution. For sake of visualization, only the three best SEGO-UTB solvers are presented, the complete obtained results are given in Figure B.6.

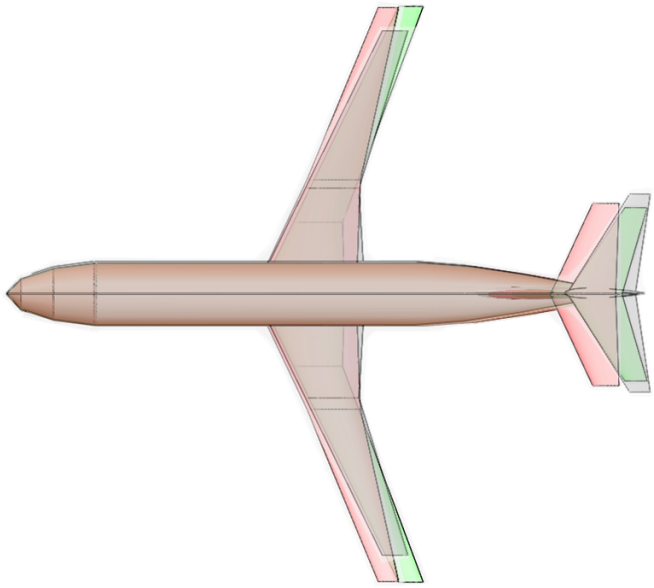
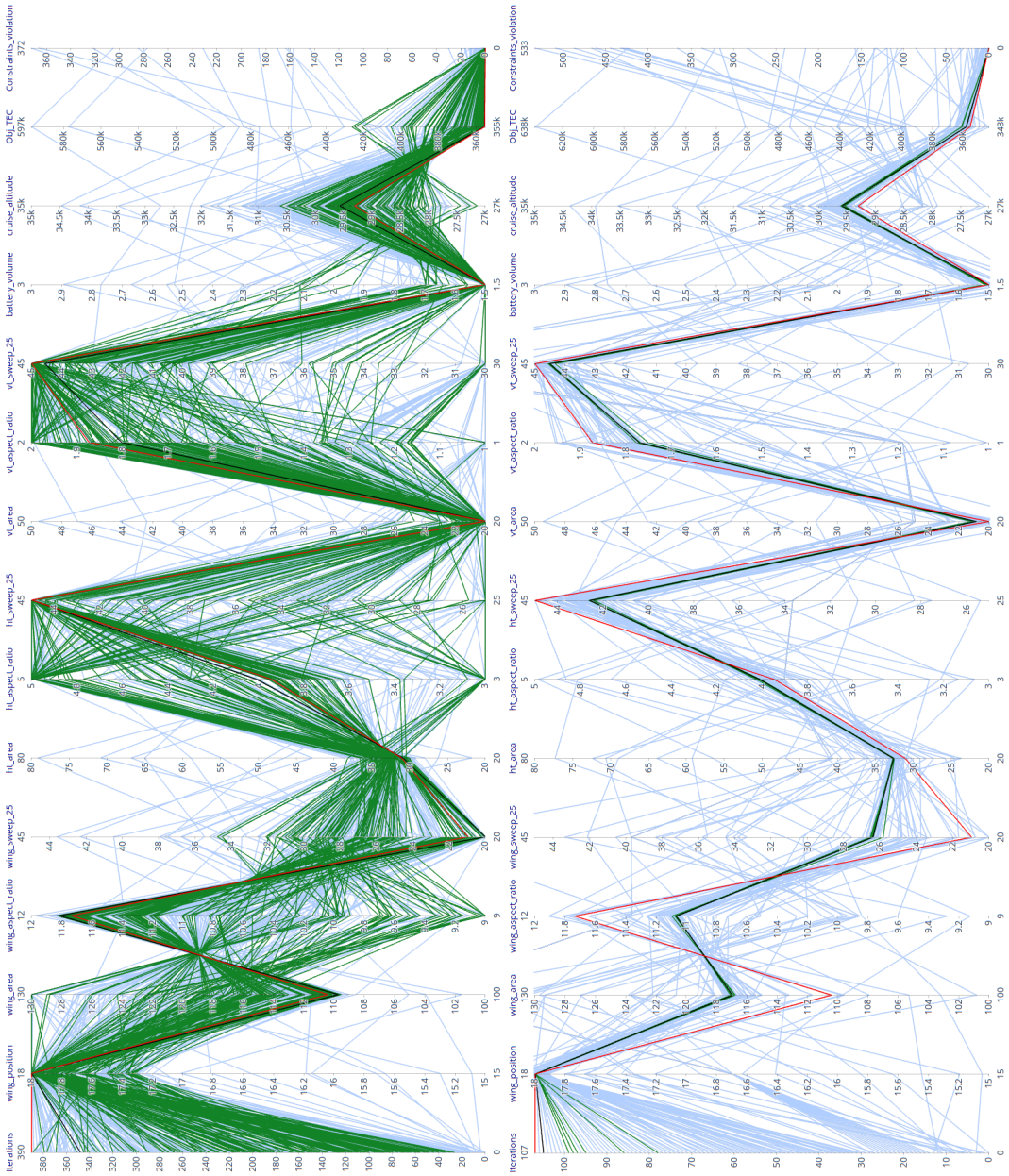


Figure 17: One of the obtained plan-forms using SEGO-like solvers (in red) and COBYLA (in green) as well as the baseline solution (in grey) of the FAST aircraft.

The obtained plan-forms (related to the FAST problem) by the SEGO-like solvers and COBYLA as well as the baseline solution are all displayed in Figure 17. The differences in the plan-forms are clearly observable. Physically, we note that both SEGO-like solvers and COBYLA solutions present a smaller sweep angle for the wing than the baseline one. The value of sweep is suggested by common design books to reduce the compressibility effects [45], but despite the transonic regime (we recall that the Mach number is 0.78), wave drag is not yet diverging, and thus the optimum goes towards a reduction of the sweep to improve aerodynamics. The minimum found by COBYLA corresponds to the best aerodynamic solution between the three; however the increased sweep leads to a greater wing mass which is penalizing. The minimum found by SEGO represents a balance between aerodynamics and mass reduction. As a consequence, the tails are reduced because of the reduction in mass and wing area (snowball effects).

In what comes next, we will use the so-called "parallel plots" to illustrate the behaviour of the tested solvers with regard to the exploration of the design space. In a parallel plot, we display the values of specific targeted data during the optimization process for a given solver (e.g., the values of the explored design variables). In our case, we depict in the parallel plots (from bottom to top) the required number of iterations to converge, the 12 design variables, the objective function value and last the constraints violation. On the top of that, we will use the red color to refer to the reference design (i.e. the best feasible design found so far by all the tested optimizers, here SEGO-UTB (τ: Cst)), in black the optimum design found by the tested solver, in green the feasible explored design, and the color blue to outline unfeasible designs. Due to the stochastic nature of our tests, we build our parallel plots using a median run on the following way: for each run of the optimizer, we store the best valid objective value. If none of the runs converges to a feasible point, we collect the minimal violation explored by the optimizer. The median run is then selected based on the stored values for all runs.

Figure 18 represents the obtained parallel plots based on the median run of the two solvers SEGO-UTB (τ: I-Log-1) and COBYLA. The parallel plots for the other optimizers are displayed in Appendix A. We note that in terms of the exploitation behaviour, represented by the convergence speed to a feasible design, SEGO-UTB (τ: I-Log-1) is clearly outperforming COBYLA. In fact, unlike COBYLA, the majority of designs obtained by SEGO-UTB (τ: I-Log-1) get feasible from the beginning of the optimization, see the iterations axis (on the bottom) of the parallel plots given by Figure 18a. The exploration behaviour is observed between iterations 250 and 390 with a majority of blue curves on the iterations axis. Apart from the horizontal and vertical tails aspect ratio values (i.e. 7^{th} and 10^{th} axis), the SEGO-UTB (τ: I-Log-1) converges to the reference design. We notice also that COBYLA is not always evaluating designs that respect the bound constraints, as shown by the designs drawn in grey on Figure 18b between iterations 15 and 30. Finally, the COBYLA best design has an optimal TEC value close to the reference one (around 5%) even if



(a) SEGO-UTB (τ: I-Log-1) (b) COBYLA

Figure 18: Parallel plots using the median run for FAST problem. In grey: the designs outside of the design space; in blue: the unfeasible designs; in green: the feasible designs; in black: the optimum; in red: the reference design.

some variables are different from the reference value as depicted in Figure 18b. However, these variables are of second importance according to the previous work of Sgueglia et al. [19]. In fact, the main driver for hybrid aircraft design is the battery volume which matches in both cases.

6. Conclusions

The SEGO solver addresses the mixed constrained optimization problems in the Bayesian optimization scope. However, it has difficulties to solve problems where the constraints are not well approximated by the GPs during the optimization process. In this paper, on the top of SEGO, we propose to use the upper trust bound while modeling the constraints to enhance the exploration of the design space. The proposed estimation combines the GP mean prediction and the associated uncertainty estimation. The included upper trust bounds on the constraints were monitored using constraints learning rates. Three different evolutions for such rates were explored; constant, decreasing and non-decreasing.

Using 29 constrained optimization problems, our proposed methods outperformed existing BO solvers (i.e., ALBO, EFI, SUR and PESC), COBYLA and NOMAD. In particular, the use of a non-decreasing logarithmic update strategy for the constraints learning rate, turns to be very useful for the SEGO framework. The good performances of the proposed solvers were confirmed on an aircraft design problem. That is why, we suggest to use the SEGO-UTB (τ: I-Log-2) as default settings for a naive user even if the choice of the constraints learning is clearly problem dependent. Indeed, an exploratory constraints learning rate must be preferred to exploitative one on highly non linear problems.

Acknowledgements

This work is part of the activities of ONERA–ISAE–ENAC joint research group.

References

[1] D. P. Raymer, Aircraft Design: A Conceptual Approach, Sixth Edition, Virginia: American Institute of Aeronautics & Astronautics, 2018.

[2] P. I. Frazier, A tutorial on Bayesian optimization, arXiv:1807.02811 .

[3] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, N. de Freitas, Taking the Human Out of the Loop: A Review of Bayesian Optimization, Proceedings of the IEEE 104 (2016) 148–175.

[4] Z. Wang, S. Jegelka, Max-Value Entropy Search for Efficient Bayesian Optimization, in: 34th International Conference on Machine Learning, vol. 70, 3627–3635, 2017.

[5] D. G. Krige, A Statistical Approach to Some Basic Mine Valuation Problems on the Witwatersrand, Journal of the Southern African Institute of Mining and Metallurgy 52 (1951) 119–139.

[6] C. E. Rasmussen, C. K. I. Williams, Gaussian Processes for Machine Learning, The MIT Press, 2006.

[7] P. Feliot, J. Bect, E. Vazquez, A Bayesian Approach to Constrained Single- and Multi-Objective Optimization, Journal of Global Optimization 67 (2017) 97–133.

[8] M. Gelbart, J. Snoek, R. Adams, Bayesian Optimization with Unknown Constraints, in: 30th Conference on Uncertainty in Artificial Intelligence, UAI 2014, 250–259, 2014.

[9] B. Letham, B. Karrer, G. Ottoni, E. Bakshy, Constrained Bayesian Optimization with Noisy Experiments, Bayesian Analysis 14 (2019) 495–519.

[10] J. Nocedal, S. J. Wright, Numerical Optimization, Springer, 2006.

[11] M. J. Sasena, P. Papalambros, P. Goovaerts, Exploration of Metamodeling Sampling Criteria for Constrained Global Optimization, Engineering optimization 34 (2002) 263–278.

[12] V. Picheny, R. B. Gramacy, S. Wild, S. Le Digabel, Bayesian Optimization under Mixed Constraints with a Slack-Variable Augmented Lagrangian, in: Advances in Neural Information Processing Systems 29, 1435–1443, 2016.

[13] R. B. Gramacy, G. A. Gray, S. Le Digabel, H. K. Lee, P. Ranjan, G. Wells, S. M. Wild, Modeling an Augmented Lagrangian for Blackbox Constrained Optimization, Technometrics 58 (2016) 1–11.

[14] D. R. Jones, M. Schonlau, W. J. Welch, Efficient Global Optimization of Expensive Black-Box Functions, Journal of Global optimization 13 (1998) 455–492.

[15] M. A. Bouhlel, N. Bartoli, R. G. Regis, A. Otsmane, J. Morlier, Efficient Global Optimization for High-Dimensional Constrained Problems by Using the Kriging Models Combined with the Partial Least Squares Method, Engineering Optimization (2018) 1–16.

[16] R. Lam, D. Allaire, K. Willcox, Multifidelity optimization using statistical surrogate modeling for non-hierarchical information sources, in: 56th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, 143, 2015.

[17] R. Lam, K. Willcox, Lookahead Bayesian optimization with inequality constraints, in: Advances in Neural Information Processing Systems, 1890–1900, 2017.

[18] R. Priem, N. Bartoli, Y. Diouane, On the Use of Upper Trust Bounds in Constrained Bayesian Optimization Infill Criteria, in: AIAA Aviation 2019 Forum, 2986, 2019.

[19] A. Sgueglia, P. Schmollgruber, N. Bartoli, O. Atinault, N. Benard, J. Morlier, Exploration and sizing of a large passenger aircraft with distributed electric ducted fans, in: 2018 AIAA Aerospace Sciences Meeting, 1745, 2018.

[20] K. Kandasamy, J. Schneider, B. Póczos, High Dimensional Bayesian Optimisation and Bandits via Additive Models, in: 32nd International Conference on Machine Learning, 295–304, 2015.

[21] N. Bartoli, T. Lefebvre, S. Dubrueil, R. Olivanti, R. Priem, N. Bons, J. R. R. A. Martins, J. Morlier, Adaptive modeling strategy for constrained global optimization with application to aerodynamic wing design, Aerospace Science and technology 90 (2019) 85–102.

[22] J. Parr, C. M. E. Holden, A. I. J. Forrester, A. J. Keane, Review of Efficient Surrogate Infill Sampling Criteria with Constraint Handling, in: 2nd International Conference on Engineering Optimization, 1–10, 2010.

[23] N. Srinivas, A. Krause, S. M. Kakade, M. Seeger, Gaussian Process Optimization in the Bandit Setting: No Regret and Experimental Design, in: 27th International Conference on Machine Learning (ICML-10), 1015–1022, 2010.

[24] M. A. Bouhlel, J. T. Hwang, N. Bartoli, R. Lafage, J. Morlier, J. R. R. A. Martins, A Python surrogate modeling framework with derivatives, Advances in Engineering Software (2019) 102662.

[25] T. P. Runarsson, X. Yao, Search biases in constrained evolutionary optimization, IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 35 (2005) 233–243.

[26] S. G. Johnson, The NLopt nonlinear-optimization package, Software available at `https://nlopt.readthedocs.io`, 2014.

[27] P. E. Gill, W. Murray, M. A. Saunders, SNOPT: An SQP algorithm for large-scale constrained optimization, SIAM review 47 (2005) 99–131.

[28] R. E. Perez, P. W. Jansen, J. R. R. A. Martins, pyOpt: a Python-based object-oriented framework for nonlinear constrained optimization, Structural and Multidisciplinary Optimization 45 (2012) 101–118.

[29] V. Picheny, A Stepwise Uncertainty Reduction Approach to Constrained Global Optimization, in: 17th on Artificial Intelligence and Statistics, vol. 33, 787–795, 2014.

[30] J. M. Hernández-Lobato, M. A. Gelbart, R. P. Adams, M. W. Hoffman, Z. Ghahramani, A General Framework for Constrained Bayesian Optimization Using Information-Based Search, Journal of Machine Learning Research (2016) 5549–5601.

[31] M. Schonlau, W. J. Welch, D. R. Jones, Global versus Local Search in Constrained Optimization of Computer Models, Lecture Notes-Monograph Series 34 (1998) 11–25.

[32] V. Picheny, D. Ginsbourger, O. Roustant, DiceOptim: Kriging-Based Optimization for Computer Experiments, Software available at `https://github.com/cran/DiceOptim`, 2014.

[33] C. Audet, J. E. Dennis Jr, Mesh adaptive direct search algorithms for constrained optimization, SIAM Journal on optimization 17 (2006) 188–217.

[34] S. Le Digabel, Algorithm 909: NOMAD: Nonlinear optimization with the MADS algorithm, ACM Transactions on Mathematical Software (TOMS) 37 (2011) 44.

[35] M. J. D. Powell, Direct search algorithms for optimization calculations, Acta numerica 7 (1998) 287–336.

[36] E. Jones, T. Oliphant, P. Peterson, et al., SciPy: Open source scientific tools for Python, Software available at `http://www.scipy.org/`, 2001–.

[37] J. M. Parr, A. J. Keane, A. I. Forrester, C. M. Holden, Infill Sampling Criteria for Surrogate-Based Optimization with Constraint Handling, Engineering Optimization 44 (2012) 1147–1166.

[38] E. Mezura-Montes, O. Cetina-Domínguez, Empirical Analysis of a Modified Artificial Bee Colony for Constrained Numerical Optimization, Applied Mathematics and Computation 218 (2012) 10943–10973.

[39] R. G. Regis, Evolutionary Programming for High-Dimensional Constrained Expensive Black-Box Optimization Using Radial Basis Functions, IEEE Transactions on Evolutionary Computation 18 (2014) 326–347.

[40] J. J. Moré, S. M. Wild, Benchmarking derivative-free optimization algorithms, SIAM Journal on Optimization 20 (2009) 172–191.

[41] Y. Diouane, S. Gratton, L. N. Vicente, Globally Convergent Evolution Strategies for Constrained Optimization, Comput. Optim. Appl. 62 (2015) 323–346.

[42] A. Sgueglia, Methodology for sizing and optimizing a Blended Wing-Body with distributed electric ducted fans, Ph.D. thesis, ISAE, 2019.

[43] A. Sgueglia, P. Schmollgruber, N. Bartoli, E. Benard, J. Morlier, J. Jasa, J. Martins, J. T. Hwang, J. Gray, Multidisciplinary design optimization framework with coupled derivative computation for hybrid aircraft, to appear in Journal of Aircraft 2020 .

[44] P. Schmollgruber, J. Bedouet, A. Sgueglia, S. Defoort, R. Lafage, N. Bartoli, Y. Gourinat, E. Benard, Use of a certification constraints module for aircraft design activities, in: 17th AIAA Aviation Technology, Integration, and Operations Conference, 3762, 2017.

[45] J. Roskam, Airplane design part I: preliminary sizing of airplanes, DAR Corporation, 4 edn., 2005.

[46] J. S. Gray, G. K. Kenway, C. A. Mader, J. R. R. A. Martins, Aero-propulsive design optimization of a turboelectric boundary layer ingestion propulsion system, in: 2018 Aviation Technology, Integration, and Operations Conference, 3976, 2018.

[47] J. Freeman, P. Osterkamp, M. W. Green, A. R. Gibson, B. T. Schiltgen, Challenges and opportunities for electric aircraft thermal management, Aircraft Engineering and Aerospace Technology: An International Journal 86 (2014) 519–524.

[48] O. Tremblay, L. A. Dessaint, Experimental validation of a battery dynamic model for EV applications, World Electric Vehicle Journal 3 (2009) 289–298.

[49] A. G. de Barros, S. C. Wirasinghe, New aircraft characteristics related to airport planning, in: 1st ATRG Conference, 25–27, 1997.

Appendix A. Academic Problems

In this appendix, we provide the academic problems. A focus is first done on the representative problems and then an overview of the 29 problem benchmark information is given.

Appendix A.1. The representative problems

An outline of the presentation of the representative constrained problems is as follows. As some of the problems share the same objective and constraints functions, we first introduce the objective functions. Then, an overview of the constraints is given. Finally, The problems definitions are provided.

Appendix A.1.1. Objective functions

The f_1 objective function is linear, f_2 is a centered and rescaled version of the *Goldstein-Price* function [12] and f_3 is the modified Branin function [22].

$$
\begin{aligned}
f_1(\boldsymbol{x}) &= \sum_{i=1}^{d} x_i \\
f_2(\boldsymbol{x}) &= \frac{\log\left[\left(1+a(4x_1+4x_2-3)^2\right)\left(30+b(8x_1-12x_2+2)^2\right)\right]-8.69}{2.43}, \text{ with} \\
a &= 75-56(x_1+x_2)+3(4x_1-2)^2+6(4x_1-2)(4x_2-2)+3(4x_2-2)^2 \\
b &= -14-128x_1+12(4x_1-2)^2+192x_2-36(4x_1-2)(4x_2-2) \\
&\quad +27(4x_2-2)^2 \\
f_3(\boldsymbol{x}) &= \left[\left(x_2-\frac{5.1x_1^2}{4\pi^2}+\frac{5x_1}{\pi}-6\right)^2+\left(10-\frac{10}{8\pi}\right)\cos(x_1)+1\right]+\frac{5x_1+25}{15}
\end{aligned}
$$

Appendix A.1.2. Constraints functions

The c_1 and c_2 constraints are the *toy* problem ones, c_3 is the centered and rescaled Branin function, c_4 is taken from Parr et al. [37], c_5 is the centered *Ackley* function, c_6 is the *Hartman* function centered and rescaled and c_7 is the constrained function of the modified Branin problem. All these functions can be

found in Picheny et al. [12], Parr et al. [22].

$$
\begin{aligned}
c_1(\boldsymbol{x}) &= 0.5\sin\left(2\pi\left(x_1^2 - 2x_2\right)\right) + x_1 + 2x_2 - 1.5 \\
c_2(\boldsymbol{x}) &= -x_1^2 - x_2^2 + 1.5 \\
c_3(\boldsymbol{x}) &= 15 - \left(15x_2 - \frac{5}{4\pi^2}\left(15x_1 - 5\right)^2 + \frac{5}{\pi}\left(15x_1 - 5\right) - 6\right)^2 \\
&\quad - 10\left(1 - \frac{1}{8\pi}\right)\cos\left(15x_1 - 5\right) \\
c_4(\boldsymbol{x}) &= 4 - \left(4 - 2.1\left(2x_1 - 1\right)^2 + \frac{\left(2x_1 - 1\right)^4}{3}\right)\left(2x_1 - 1\right)^2 - (2x_1 - 1)(2x_2 - 1) \\
&\quad - 16\left(x_2^2 - x_2\right)\left(2x_2 - 1\right)^2 - 3\sin\left(12(1 - x_1)\right) - 3\sin\left(12(1 - x_2)\right) \\
c_5(\boldsymbol{x}) &= 20\exp\left(-0.2\sqrt{\frac{1}{4}\sum_{i=1}^{4}\left(3x_i - 1\right)^2}\right) + \exp\left(\frac{1}{4}\sum_{i=1}^{4}\cos\left(2\pi(3x_i - 1)\right)\right) \\
&\quad - 17 - \exp\left(1\right) \\
c_6(\boldsymbol{x}) &= \frac{1}{0.8387}\left[-1.1 + \sum_{i=1}^{4} C_i \exp\left(-\sum_{j=1}^{4} a_{ij}(x_j - p_{ij})^2\right)\right] \\
c_7(\boldsymbol{x}) &= 6 - \left(4 - 2.1\overline{x}_1^2 + \frac{\overline{x}_1^4}{3}\right)\overline{x}_1^2 - \overline{x}_1\overline{x}_2 - \left(4\overline{x}_2^2 - 4\right)\overline{x}_2^2 - 3\sin\left(6(1 - \overline{x}_1)\right) \\
&\quad - 3\sin\left(6(1 - \overline{x}_2)\right)
\end{aligned}
$$

with:

$$
\boldsymbol{a} = \begin{bmatrix} 10.00 & 0.05 & 3.00 & 17.00 \\ 3.00 & 10.00 & 3.50 & 8.00 \\ 17.00 & 17.00 & 1.70 & 0.05 \\ 3.50 & 0.10 & 10.00 & 10.00 \end{bmatrix},\ \boldsymbol{p} = \begin{bmatrix} 0.131 & 0.232 & 0.234 & 0.404 \\ 0.169 & 0.413 & 0.145 & 0.882 \\ 0.556 & 0.830 & 0.352 & 0.873 \\ 0.012 & 0.373 & 0.288 & 0.574 \end{bmatrix},
$$

$$
\boldsymbol{C} = \begin{bmatrix} 1.0 & 1.2 & 3.0 & 3.2 \end{bmatrix},\ \overline{x}_1 = \frac{x_1 - 2.5}{7.5},\ \overline{x}_2 = \frac{x_2 - 7.5}{7.5}
$$

Appendix A.1.3. Problems

The four representative problems and the *modified Branin problem with equality constraints* (MBE) are expressed as follows:

$$\text{(MB):}\ \min_{\boldsymbol{x}\in\Omega_3} f_3(\boldsymbol{x})\ \ \text{s.t.}\ \ c_7(\boldsymbol{x}) \leq 0 \tag{A.1}$$

$$\text{(LSQ):}\ \min_{\boldsymbol{x}\in\Omega_1} f_1(\boldsymbol{x})\ \ \text{s.t.}\ \ c_1(\boldsymbol{x}) \geq 0,\ c_2(\boldsymbol{x}) \geq 0 \tag{A.2}$$

$$\text{(GBSP):}\ \min_{\boldsymbol{x}\in\Omega_1} f_2(\boldsymbol{x})\ \ \text{s.t.}\ \ c_1(\boldsymbol{x}) \geq 0,\ c_2(\boldsymbol{x}) = 0,\ c_3(\boldsymbol{x}) = 0 \tag{A.3}$$

$$\text{(LAH):}\ \min_{\boldsymbol{x}\in\Omega_2} f_1(\boldsymbol{x})\ \ \text{s.t.}\ \ c_5(\boldsymbol{x}) \leq 0,\ c_6(\boldsymbol{x}) = 0 \tag{A.4}$$

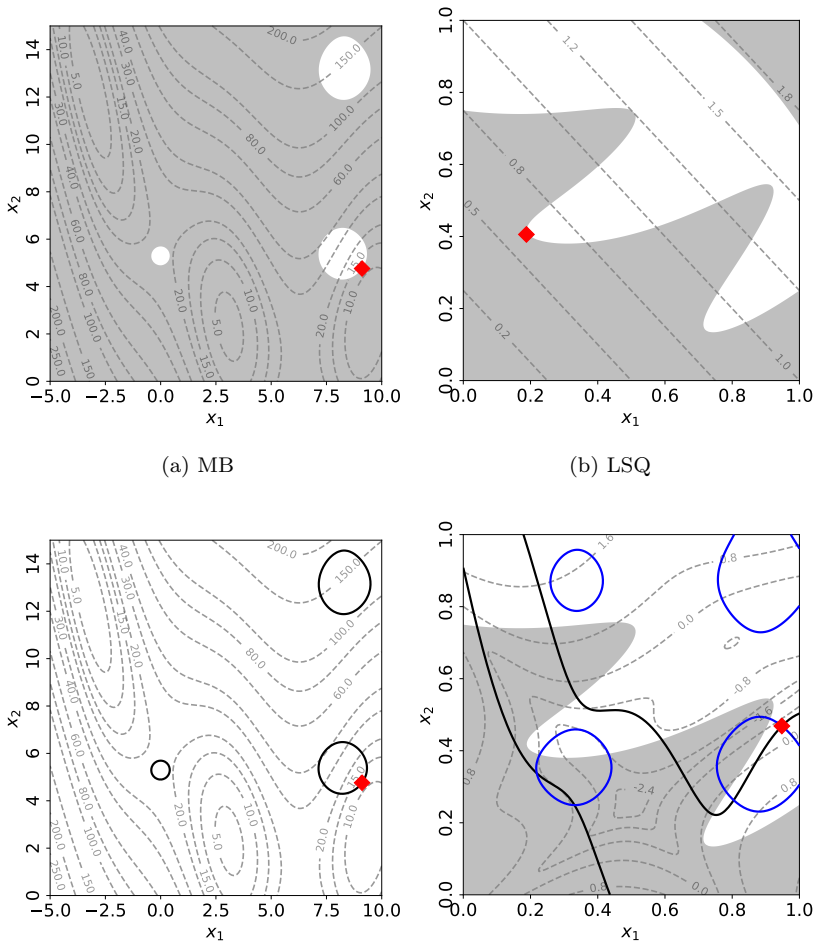
$$\text{(MBE):}\ \min_{\boldsymbol{x}\in\Omega_3} f_3(\boldsymbol{x})\ \ \text{s.t.}\ \ c_7(\boldsymbol{x}) = 0 \tag{A.5}$$

with: $\Omega_1 = [0,1]^2$, $\Omega_2 = [0,1]^4$ and $\Omega_3 = [-5,10] \times [0,15]$.

The two dimensional problems are drawn in Figure A.1 to a better understanding of the addressed challenges. Figure A.1a shows that the MB problem has disjoint feasible domains and a multimodal objective function. The LSQ problem is interesting because of the non convexity of the feasible domain (see Figure A.1b). The MBE equality constrained problem has a multimodal objective function and three disjoint feasible domains have been drawn in Figure A.1c. Then, Figure A.1d displays the GBSP mixed constrained problem. It is the most challenging one as it gathers a multimodal objective function, a non convex feasible domain concerning the inequality constraint and disjoint feasible domains for equality ones. The resulting feasible domain is thus restricted to only two points. Finally, the LAH problem cannot be displayed as it is a four dimensional problem.

Appendix A.2. The 29 problem benchmark

The 29 problem benchmark is composed of well known optimization problems from Picheny et al. [12], Parr et al. [22], Mezura-Montes and Cetina-Domínguez [38], Regis [39]. We picked the problems between 2 to 10 design variables with equality, inequality and mixed constraints. These problems are also segregated into two categories:



(a) MB (b) LSQ

(c) MBE (d) GBSP

Figure A.1: Representation of the problems. Dashed curves: contour plot of the objective function ; grey area: unfeasible domain for inequality constraints ; filled curves: feasible domain for equality constraints ; red square: global optimum.

- WNLC, meaning that all their constraints are linear or quadratic.
- HNLC, meaning at least one of the constraints is not linear or quadratic.

All these information are detailed for each problem in Table A.1.

Table A.1: Some of the features of the tested constrained optimization problem.

Problem Name	**Nb. of Variables**	**Nb. of Eq. Cst.**	**Nb. of Ieq. Cst.**	**Linear Type**	**Ref. Value**	**Problem Name**	**Nb. of Variables**	**Nb. of Eq. Cst.**	**Nb. of Ieq. Cst.**	**Linear Type**	**Ref. Value**
G03	10	1	0	WNLC	-1.000	G04	5	0	6	WNLC	$-3.067 \cdot 10^4$
G05	4	3	2	HNLC	5126	G06	2	0	2	WNLC	-6962
G07	10	0	8	WNLC	24.23	G08	2	0	2	WNLC	$-9.583 \cdot 10^{-2}$
G09	7	0	4	WNLC	680.6	G10	8	0	6	WNLC	7049
G11	2	1	0	WNLC	0.750	G12	3	0	1	WNLC	-1.000
G13	5	3	0	HNLC	$2.201 \cdot 10^{-3}$	G14	10	3	0	WNLC	-47.71
G15	3	2	0	WNLC	961.7	G16	5	0	38	HNLC	-1.918
G17	6	4	0	WNLC	8864	G18	9	0	13	WNLC	-0.8661
G21	7	5	1	WNLC	193.8	G23	9	4	2	WNLC	-400.1
G24	2	0	2	HNLC	-6.031	WB4	4	0	6	HNLC	0.4734
GBSP	2	2	1	HNLC	-0.5252	GTCD	4	0	1	HNLC	$2.965 \cdot 10^6$
Hesse	6	0	6	WNLC	-310.0	LAH	4	1	1	HNLC	$5.176 \cdot 10^{-2}$
LSQ	2	0	2	HNLC	0.600	SR7	7	0	11	HNLC	2994
MB	2	0	1	HNLC	12.00	MBE	2	1	0	HNLC	12.00
PVD4	4	0	3	HNLC	5809						

Appendix B. Additional results

In this appendix, we comment the results on the four test problems from Picheny et al. [12], Parr et al. [22], the 29 problem benchmark and the FAST problem for the non-decreasing and decreasing constraints learning rate strategies. For more information on the tests plan and methodology, see Sections 4 and 5.

Appendix B.1. The representative problems

Figure B.1 shows the averaged best valid value for non-decreasing number of evaluations of the GBSP problem using the constraints violation of 10^{-2} and 10^{-4}. Using both constraints violation, it appears that SEGO-UTB (τ: D-Exp-2) (resp. (τ: I-Exp-2)) shows the best compromise for the decreasing (resp. non-decreasing) constraints learning rate strategy.

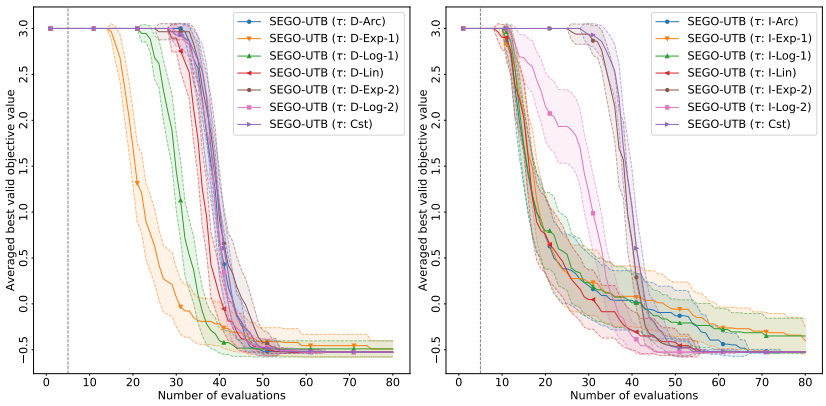
For the LAH problem, all the non-decreasing and decreasing learning rates are performing almost the same and are converging in less than 40 iterations as shown by Figure B.2. We choose arbitrarily SEGO-UTB (τ: D-Exp-2) and resp. SEGO-UTB (τ: I-Exp-2) as references constraints learning rate strategies for the LAH problem.

SEGO-UTB (τ: D-Exp-2) (resp. SEGO-UTB (τ: I-Exp-2)) provides the best compromise on LSQ problem as implies by Figure B.3. Indeed, they both converge to the minimum value using the two levels of constraints violation and the standard deviation tend faster to zero than the other strategies.

Figure B.4 provides the results for the non-decreasing and decreasing constraints learning rate strategies of SEGO-UTB for the MB problem using the two constraints violation 10^{-2} and 10^{-4}. Again, SEGO-UTB (τ: D-Exp-2) and SEGO-UTB (τ: I-Exp-2) are a good compromise considering the two constraints violations. Note that none of the non-decreasing constraints learning rate are converging for the MB problem.

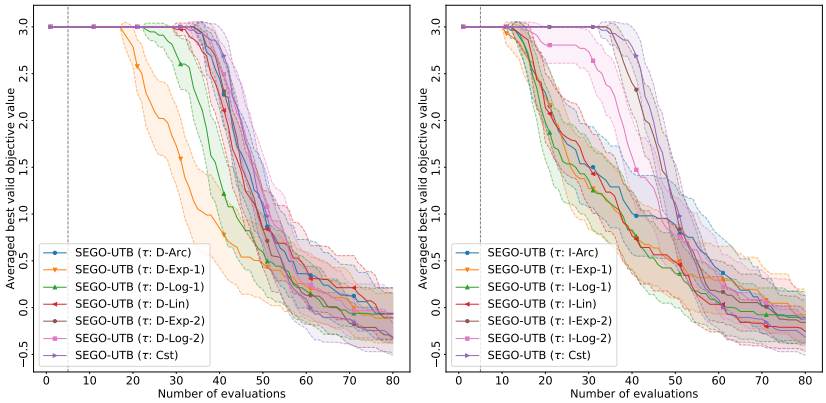
Appendix B.2. The 29 problem benchmark

Figure B.5 shows the data profiles for the non-decreasing and decreasing constraints learning rate strategies using the two levels of constraints violation



(a) Decreasing constraints learning rate strategies using $\epsilon_c = 10^{-2}$.

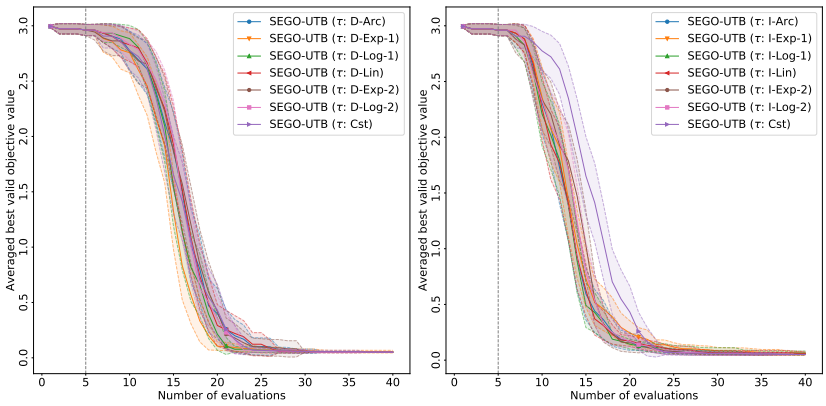
(b) Non-decreasing constraints learning rate strategies using $\epsilon_c = 10^{-2}$.



(c) Decreasing constraints learning rate strategies using $\epsilon_c = 10^{-4}$.

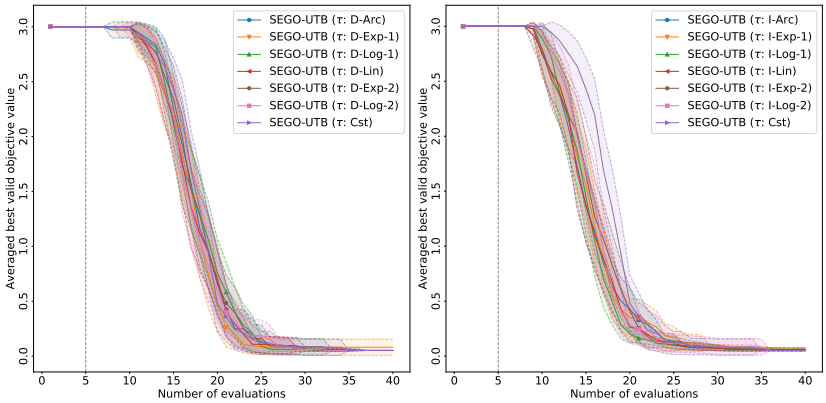
(d) Non-decreasing constraints learning rate strategies using $\epsilon_c = 10^{-4}$.

Figure B.1: Convergence plots for the GBSP problem, considering the two levels of constraints violation 10^{-4} and 10^{-2}. The vertical grey-dashed line outlines the number of points in the initial DoEs.



(a) Decreasing constraints learning rate strategies using $\epsilon_c = 10^{-2}$.

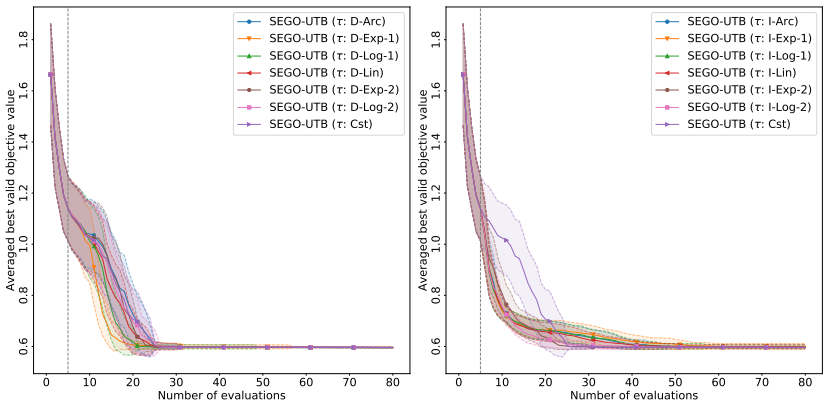
(b) Non-decreasing constraints learning rate strategies using $\epsilon_c = 10^{-2}$.



(c) Decreasing constraints learning rate strategies using $\epsilon_c = 10^{-4}$.

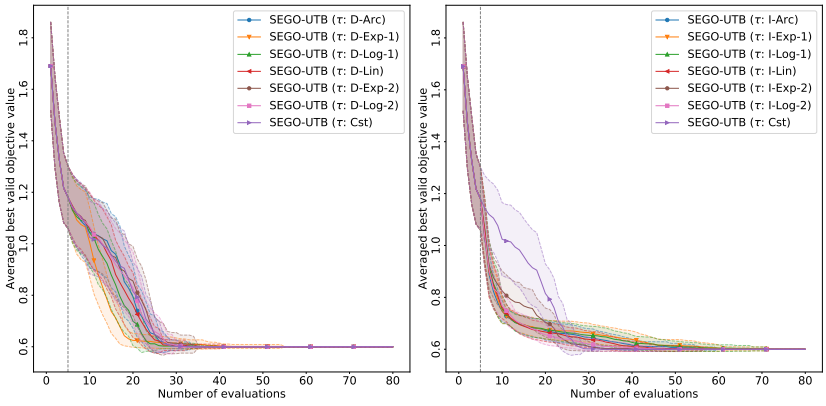
(d) Non-decreasing constraints learning rate strategies using $\epsilon_c = 10^{-4}$.

Figure B.2: Convergence plots for the LAH problem, considering the two levels of constraints violation 10^{-4} and 10^{-2}. The vertical grey-dashed line outlines the number of points in the initial DoEs.



(a) Decreasing constraints learning rate strategies using $\epsilon_c = 10^{-2}$.

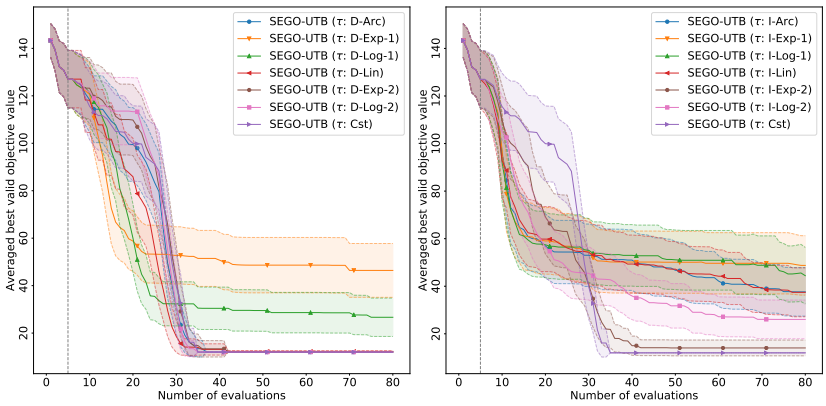
(b) Non-decreasing constraints learning rate strategies using $\epsilon_c = 10^{-2}$.



(c) Decreasing constraints learning rate strategies using $\epsilon_c = 10^{-4}$.

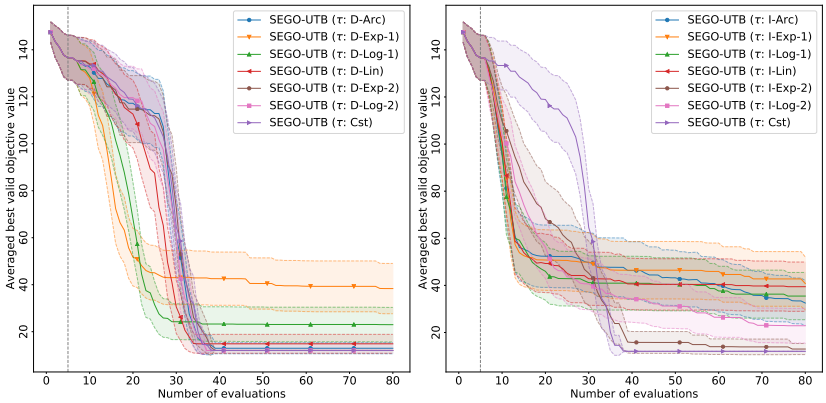
(d) Non-decreasing constraints learning rate strategies using $\epsilon_c = 10^{-4}$.

Figure B.3: Convergence plots for the LSQ problem, considering the two levels of constraints violation 10^{-4} and 10^{-2}. The vertical grey-dashed line outlines the number of points in the initial DoEs.



(a) Decreasing constraints learning rate strategies using $\epsilon_c = 10^{-2}$.

(b) Non-decreasing constraints learning rate strategies using $\epsilon_c = 10^{-2}$.

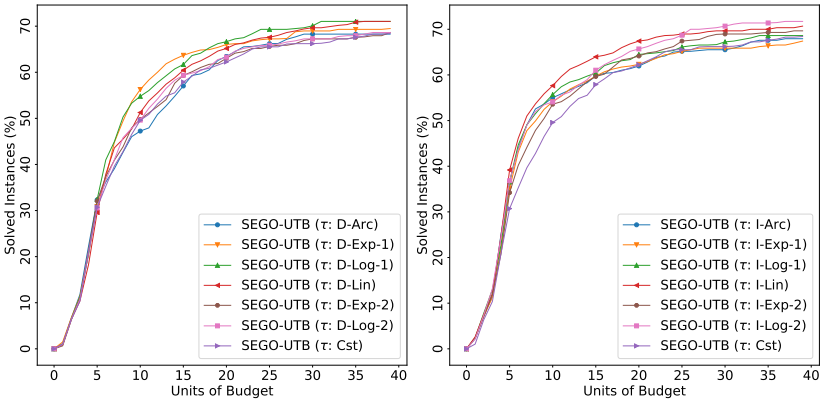


(c) Decreasing constraints learning rate strategies using $\epsilon_c = 10^{-4}$.

(d) Non-decreasing constraints learning rate strategies using $\epsilon_c = 10^{-4}$.

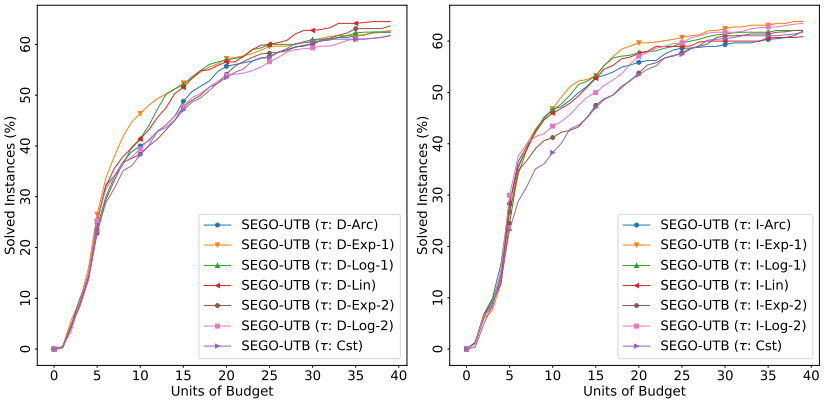
Figure B.4: Convergence plots for the MB problem, considering the two levels of constraints violation 10^{-4} and 10^{-2}. The vertical grey-dashed line outlines the number of points in the initial DoEs.

10^{-2} and 10^{-4}. Concerning the decreasing strategies, one can clearly see that SEGO-UTB (τ: D-Exp-1) is the best compromise as it solve 55% (resp. 65%) of the instances using $\epsilon_c = 10^{-2}$ (resp. $\epsilon_c = 10^{-4}$). SEGO-UTB (τ: I-Log-1) offers the best performances on the both constraints violation for the non decreasing strategies.



(a) Decreasing constraints learning rate strategies using $\epsilon_c = 10^{-2}$.

(b) Non-decreasing constraints learning rate strategies using $\epsilon_c = 10^{-2}$.



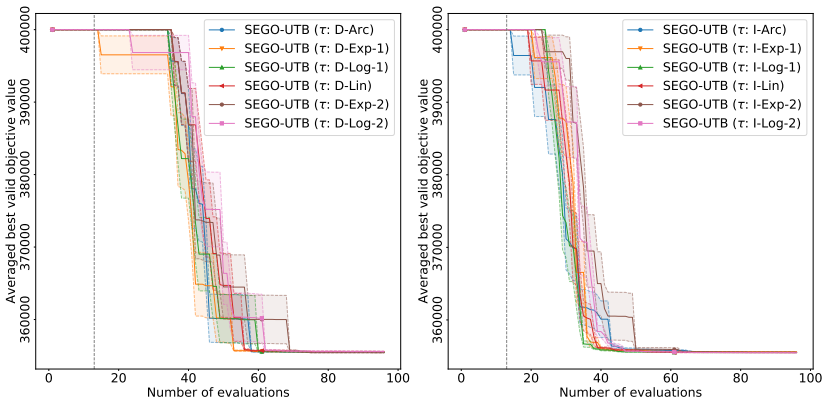
(c) Decreasing constraints learning rate strategies using $\epsilon_c = 10^{-4}$.

(d) Non-decreasing constraints learning rate strategies using $\epsilon_c = 10^{-4}$.

Figure B.5: Data Profiles for all the tested problems, considering the two levels of constraints violation 10^{-4} and 10^{-2}.

Appendix B.2.1. The FAST test case

For the FAST test case, the non-decreasing and decreasing constraints learning rates performance are given by Figure B.6. We notice that SEGO-UTB (τ: D-



(a) Decreasing constraints learning rate strategies.

(b) Non-decreasing constraints learning rate strategies.

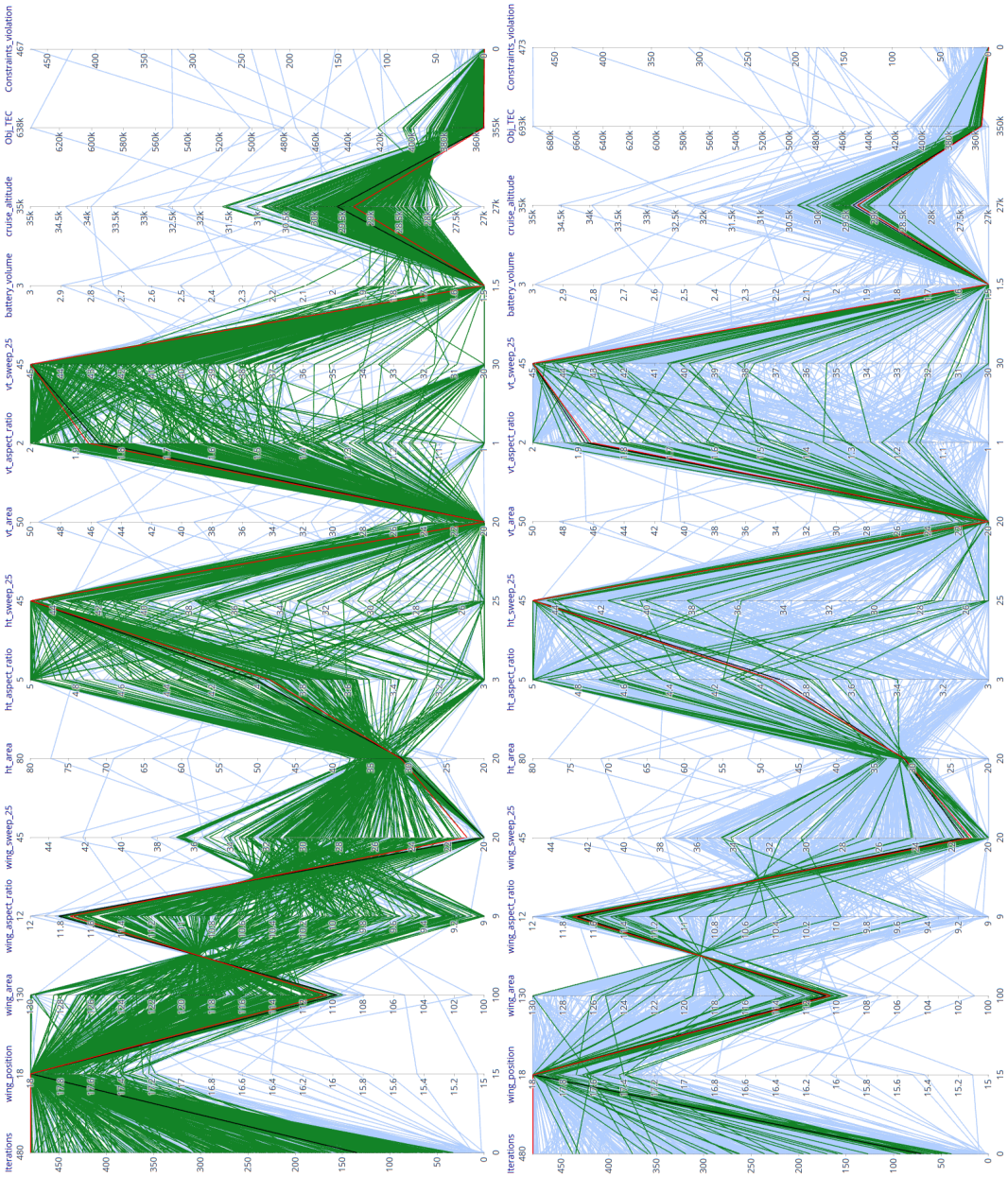
Figure B.6: Convergence plots for the FAST problem. The vertical grey-dashed line outlines the number of points in the initial DoEs.

Exp-1) is performing the best for the decreasing behaviors as it converges the fastest to the minimum value. For the non-decreasing behaviors, SEGO-UBT (τ: I-Log-1) is providing the best results.

Last, the parallel plots of the median run of SEGO, SEGO-UTB (τ: Cst), SEGO-UTB (τ: D-Exp-1) and NOMAD are introduced in Figures B.7 and B.8." B.8.

First, NOMAD is not able to find any feasible point as implied by Figure B.8b and seems to get stuck in a zone with a violation around 150. Then, SEGO-UTB (τ: Cst), drawn in Figure B.7b, highlights the exploration behavior of this constraints learning rate with an important number of unfeasible designs all along the optimization. This behavior allows the optimizer to detect the best feasible design which is very close to the reference optimum. Figure B.7a lastly displays that SEGO focuses on the exploitation of the constraints as demonstrated by

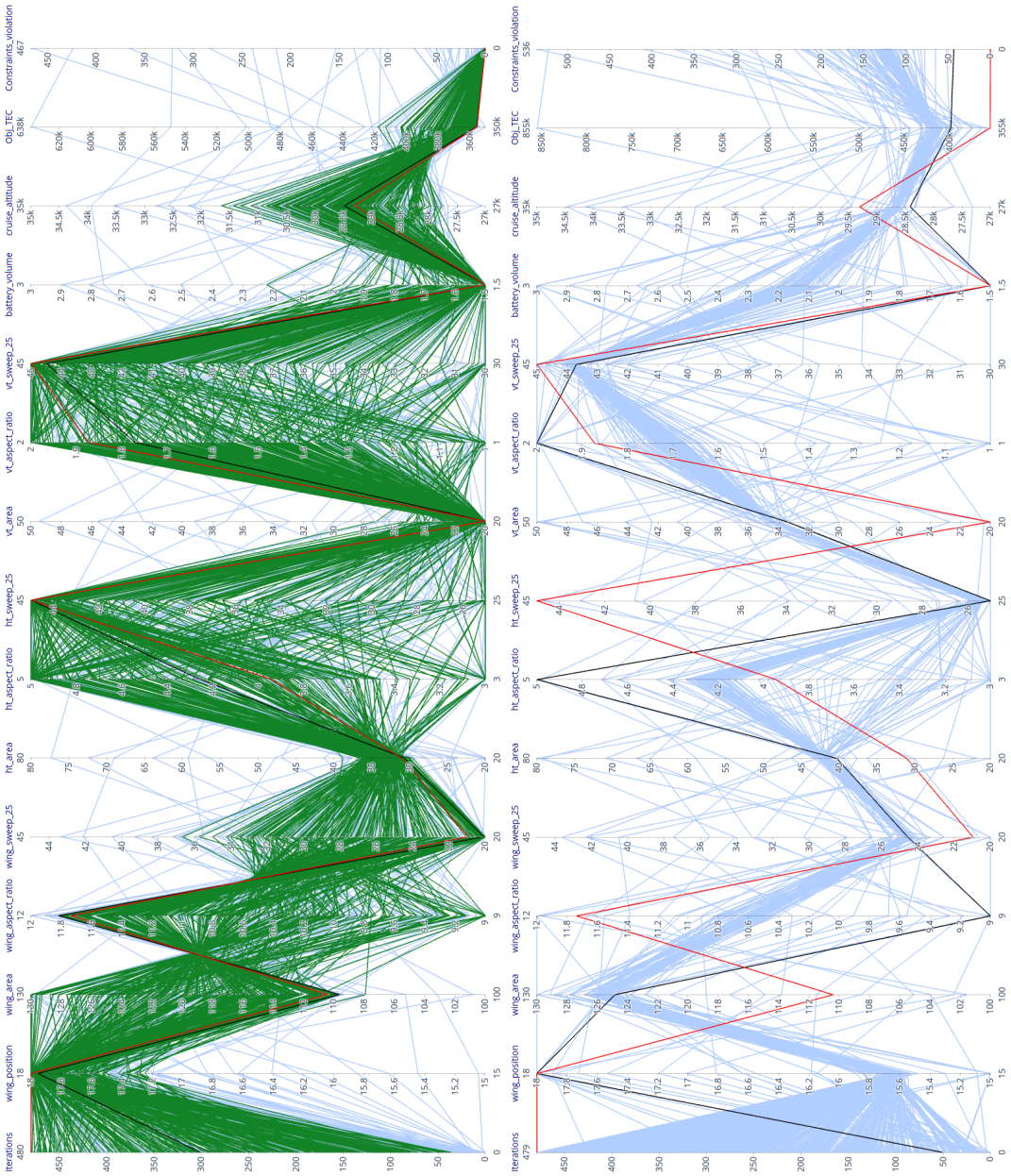
the large number of feasible design points. The optimum is thus quickly obtain.



(a) SEGO

(b) SEGO-UTB (τ: Cst)

Figure B.7: Parallel plots of the median runs for FAST problem. In grey: the designs outside of the design space; in blue: the unfeasible designs; in green: the feasible designs; in black: the optimum; in red: the reference design.



(a) SEGO-UTB (τ: D-Exp-1) (b) NOMAD

Figure B.8: Parallel plots of the median runs for FAST problem. In grey: the designs outside of the design space; in blue: the unfeasible designs; in green: the feasible designs; in black: the optimum; in red: the reference design.