Les Cahiers du GERAD ISSN: 0711–2440

Bayesian optimization with hidden constraints for aircraft design

A. Tfaily, Y. Diouane, N. Bartoli, M. Kokkolaras

G–2024–10

January 2024

La collection *Les Cahiers du GERAD* est constituée des travaux de recherche menés par nos membres. La plupart de ces documents de travail a été soumis à des revues avec comité de révision. Lorsqu'un document est accepté et publié, le pdf original est retiré si c'est nécessaire et un lien vers l'article publié est ajouté.

Citation suggérée : A. Tfaily, Y. Diouane, N. Bartoli, M. Kokkolaras (Janvier 2024). Bayesian optimization with hidden constraints for aircraft design, Rapport technique, Les Cahiers du GERAD G–2024–10, GERAD, HEC Montréal, Canada.

Avant de citer ce rapport technique, veuillez visiter notre site Web (https://www.gerad.ca/fr/papers/G-2024-10) afin de mettre à jour vos données de référence, s'il a été publié dans une revue scientifique.

The series *Les Cahiers du GERAD* consists of working papers carried out by our members. Most of these pre-prints have been submitted to peer-reviewed journals. When accepted and published, if necessary, the original pdf is removed and a link to the published article is added.

Suggested citation: A. Tfaily, Y. Diouane, N. Bartoli, M. Kokkolaras (January 2024). Bayesian optimization with hidden constraints for aircraft design, Technical report, Les Cahiers du GERAD G–2024–10, GERAD, HEC Montréal, Canada.

Before citing this technical report, please visit our website (https://www.gerad.ca/en/papers/G-2024-10) to update your reference data, if it has been published in a scientific journal.

La publication de ces rapports de recherche est rendue possible grâce au soutien de HEC Montréal, Polytechnique Montréal, Université McGill, Université du Québec à Montréal, ainsi que du Fonds de recherche du Québec – Nature et technologies.

Dépôt légal – Bibliothèque et Archives nationales du Québec, 2024
– Bibliothèque et Archives Canada, 2024

The publication of these research reports is made possible thanks to the support of HEC Montréal, Polytechnique Montréal, McGill University, Université du Québec à Montréal, as well as the Fonds de recherche du Québec – Nature et technologies.

Legal deposit – Bibliothèque et Archives nationales du Québec, 2024
– Library and Archives Canada, 2024

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

Bayesian optimization with hidden constraints for aircraft design

Ali Tfaily [a, b]

Youssef Diouane [c]

Nathalie Bartoli [d]

Michael Kokkolaras [a]

[a] *Department of Mechanical Engineering McGill University, Canada*

[b] *Advanced Product Development, Bombardier, Canada*

[c] *Department of Mathematical and Industrial Engineering, Polytechnique Montréal, Canada*

[d] *ONERA/DTIS, Université de Toulouse, France*

ali.tfaily@mail.mcgill.ca
youssef.diouane@polymtl.ca
nathalie.bartoli@onera.fr
michael.kokkolaras@mcgill.ca

January 2024
Les Cahiers du GERAD
G–2024–10

Copyright © 2024 Tfaily, Diouane, Bartoli, Kokkolaras

Les textes publiés dans la série des rapports de recherche *Les Cahiers du GERAD* n'engagent que la responsabilité de leurs auteurs. Les auteurs conservent leur droit d'auteur et leurs droits moraux sur leurs publications et les utilisateurs s'engagent à reconnaître et respecter les exigences légales associées à ces droits. Ainsi, les utilisateurs:

- Peuvent télécharger et imprimer une copie de toute publication du portail public aux fins d'étude ou de recherche privée;
- Ne peuvent pas distribuer le matériel ou l'utiliser pour une activité à but lucratif ou pour un gain commercial;
- Peuvent distribuer gratuitement l'URL identifiant la publication.

Si vous pensez que ce document enfreint le droit d'auteur, contactez-nous en fournissant des détails. Nous supprimerons immédiatement l'accès au travail et enquêterons sur votre demande.

The authors are exclusively responsible for the content of their research papers published in the series *Les Cahiers du GERAD*. Copyright and moral rights for the publications are retained by the authors and the users must commit themselves to recognize and abide the legal requirements associated with these rights. Thus, users:

- May download and print one copy of any publication from the public portal for the purpose of private study or research;
- May not further distribute the material or use it for any profit-making activity or commercial gain;
- May freely distribute the URL identifying the publication.

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Abstract : A challenge in aircraft design optimization is the presence of non-computable, so-called hidden, constraints that do not return a value in certain regions of the design space. In this paper, we present a novel method to handle hidden constraints in aircraft conceptual design using Bayesian optimization. The method entails modifying a portion of the acquisition function of a Bayesian optimization formulation using supervised machine learning classifiers. The proposed approach reduces the effect of classifiers on exploration, therefore allowing the optimization algorithm to consider regions of the design space where previous information is not available. In addition, we consider different classifiers for handling hidden constraints. We demonstrate the proposed method using two simulation-based aircraft design optimization problems related to landing gear sizing and aircraft performance. The obtained results show an improvement of the objective function with fewer function evaluations.

Keywords: Bayesian optimization, expected improvement, hidden constraints, simulation failure, machine learning classification, aircraft design

Acknowledgements: The authors would like to thank Jasveer Singh and Hugo Gagnon from Bombardier for their insights and support in setting up the industrial aircraft conceptual design problems. We also thank Rémy Priem for early discussions about topics in this work. The first and last authors are grateful for the partial support of this work by NSERC grant RGPIN 436193–18; this support does not constitute an endorsement of the opinions expressed in this paper.

1 Introduction

Aircraft development is a complex process that involves significant investment over multi-year design phases. This necessitates a strict approach for decision making that minimizes errors and the need for design re-work. The first of an aircraft's design phases is named the conceptual design phase and entails selecting the aircraft configuration and the major design parameters. It is estimated that about 70% of the projected life-cycle cost of an aircraft can be committed based on design decisions taken during the conceptual design stage of the design process [14, 34]. Therefore, aircraft manufacturers rely heavily on tools to simulate and improve an aircraft's conceptual design and have been increasingly bringing disciplines typically done in later design stages into the early conceptual design phase [9, 17, 28, 39]. In addition to adding simulation disciplines to early design phases, aircraft designers have relied on optimization algorithms since the 1970's to aid in solving problems within specific aircraft disciplines or to address overall multidisciplinary problems by means of multidisciplinary design optimization (MDO) [40]. MDO algorithms have been extensively studied and exploited on aircraft design applications, for example, for solving complex aero-structural interactions of complete aircraft configurations captured using high-fidelity models [32], or for solving conceptual design problems using lower-fidelity models that address a large number of disciplines [28]. Typical optimizations in aircraft design involve failed (crashed) simulations that prevent the completion of an optimization or its convergence. A failed simulation in an optimization is defined as a simulation that terminates unexpectedly resulting in an error in the outcomes of the optimization. In this work, we refer to these failed simulations in an optimization as hidden constraints. Hidden are not explicitly known to an optimization solver as per [23]. In aircraft design optimization, such simulation failures can happen due to several reasons ranging from failure of an aerodynamic solver to converge [25, 37] to limitations in black box simulations that cause crashes such as engine performance evaluation [8].

Surrogate-based optimization (SBO) refers to a class of methods where typically lower-fidelity physics- or data-based models are used in lieu of higher-fidelity models under the premise that the former are less computationally expensive and/or smoother than the latter. Model surrogates can be constructed during the optimization process. In this work, we consider the Bayesian optimization (BO) paradigm where Gaussian processes (GP) are used to model the objective and constraint functions. BO has become a popular method for solving optimization problems in aerospace engineering design [18, 21, 30, 35].

Handling hidden constraints in BO algorithms has been identified and investigated recently in [2, 5, 15, 24, 33, 41]. A hidden constraint typically appears when simulations within an optimization crash which makes them not quantifiable and difficult to handle. [24], used a random forest classifier to calculate a feasible probability and integrated the classifier within an expected improvement (EI) acquisition function. In [33], the authors extended a method developed by [7] to handle hidden constraints within BO. The authors use a least-squares support vector machine technique for classification of both known and hidden constraints which is used to model arly design phases, aircraft designers have relied on optimization algorithms since the 1970's to aid in solving problems within specific aircraft disciplines or to address overall multidisciplinary problems by means of multidisciplinary design optimization (MDO) [40]. MDO algorithms have been extensively studied and exploited on aircraft design applications, for example, for solving complex aero-structural interactions of complete aircraft configurations captured using high-fidelity models [32], or for solving conceptual design problems using lower-fidelity models that address a large number of disciplines [28]. Typical optimizations in aircraft design involve failed (crashed) simulations that prevent the completion of an optimization or its convergence. A failed simulation in an optimization is defined as a simulation that terminates unexpectedly resulting in an error in the outcomes of the optimization. In this work, we refer to these failed simulations in an optimization as hidden constraints. Hidden are not explicitly known to an optimization solver as per [23]. In aircraft design optimization, such simulation failures can happen due to several reasons ranging from failure of an aerodynamic solver to converge [25, 37] to limitations in black box simulations that cause crashes such as engine performance evaluation [8]. the boundary of the feasible design space of an efficient global optimization problem. [15] proposed a framework to perform BO under hidden constraints using a probabilistic approach where constraint satisfaction can be determined by meeting a probability of feasibility threshold. In [2], the author presents a sequential BO framework for problems that are undefined or partially outside the feasible region. The framework uses a support vector machine classification method to estimate the boundary of the feasible design space which is then used to construct surrogate models of the objective function. [41] proposed the use of an external classifier using Gaussian processes that determines a probability of feasibility. The calculated probability is then used to condition the acquisition function directly similar to the approach proposed in [24]. [5] developed a Gaussian process classifier and applied it on a modified EI acquisition function. The authors then provided proof of global convergence of the approach. [4] proposed in [4] the use of k-nearest neighbors classifiers to build surrogate models that guide a mesh adaptive direct search optimization algorithm. In this paper, we propose a method to handle hidden constraints and present its application on industrial test cases.

The paper is organized as follows. Section 2 presents hidden constraints in aircraft design optimization and two aircraft design optimization applications with hidden constraints. Section 3 describes the following: BO algorithm with and without hidden constraints, the proposed acquisition function to handle hidden constraints, supervised machine learning (ML) classifiers modeled to represent hidden constraints, and an analytical illustration example. Aircraft design problems results are presented in Section 4 where we compare optimization results of the proposed acquisition function with existing methods using several types of ML classifiers. Conclusions and perspectives are given in Section 5.

2 Hidden constraints in aircraft conceptual design

Hidden constraints in aircraft design optimization are dependant on the type of problem to be solved but can be generalized into two categories: (1) inability of models to generate solutions, (2) failure in simulation models due to physics-based limitations, bugs in the models, or architecture and software implementation of the models. An example of the first category is an aerodynamic optimization where the CFD solver may not converge due to complex flow fields and geometries therefore returning an error to the optimizer [25]. Another similar example would be a computer aided design (CAD) modeler's or mesh generator's inability to create models for the aerodynamic solver thus causing a failure [13]. The second category applies to simulation codes where failures occur due to either errors in a black box code or regions of the design space that cannot return a value due to the physics of the problem at hand. An example of such is presented in [11] when performing an optimization for an aircraft environment control system. In the considered problem in [11], certain designs can lead to supersonic solutions for which the values of temperatures and pressures predicted by the environment control system model are considered as simulation failures. We consider such models as non-robust models since known constraints should be created to prevent simulation failure. We note however, that there could also be a third category of hidden constraints which could be driven by other reasons for simulation failures that are considered random such as infrastructure related issues. An example we encountered where communication issues between a cloud computing platform and simulation models caused failures of some iterations of an optimization. Such failures must be addressed differently from what we consider as hidden constraints in this paper, that is by solving the issues causing such failures. We do not address the first category of hidden constraint problems in this paper (i.e., inability of models to generate solutions such as CFD-based aerodynamic design optimization) as there is a dedicated field of research to increase solver robustness as discussed in [25]. We target instead the second category of hidden constraints, i.e., black box simulation models with failure, and consider aircraft applications of this category.

In this paper, we use a simulation-based aircraft conceptual design optimization problem as the application to demonstrate hidden constraints by leveraging Bombardier's multilevel multidisciplinary optimization framework [28]. The chosen aircraft is based on the Bombardier Research Aircraft (BRAC) discussed in [30, 31], see Figure 1. The problem is a minimization of aircraft maximum takeoff weight

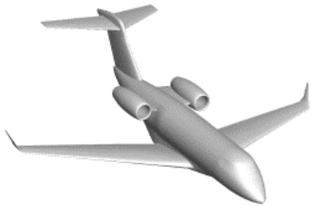


Figure 1: 3D model of the BRAC aircraft used in the two industrial application problems.

(MTOW) using 12 design variables and subject to 12 inequality constraints. The optimization problem is formulated as follows:

$$\begin{cases} \underset{x \in \Omega}{\text{minimize}} & \text{MTOW}(x) \\ \text{subject to} & c_i(x) \leq 0, \; i = 1, \ldots, 12, \end{cases} \tag{1}$$

where MTOW represents the aircraft maximum takeoff weight, $x \in \Omega$, defined as the design space $\subset \mathbb{R}^{12}$, is the vector of the design variables (see Table 1 for a detailed description) which are all bounded, and $c_i(x)$ $i = 1, \ldots, 12$ are the inequality constraints described in Table 2.

Table 1: A list of the design variables related to the aircraft conceptual design.

Design variables	Description
x_1	Rubber engine scaling factor
x_2	Wing aspect ratio
x_3	Wing area
x_4	Wing trailing edge sweep
x_5 and x_6	Wing rear spar chord-wise location
x_7	Wing sweep
x_8	Wing taper ratio
$x_9, \ldots, x_{12}$	Wing thickness-to-chord ratios

Table 2: A list of the aircraft conceptual design problem constraints.

Constraint	Description
$c_1(x)$	Balanced field length
$c_2(x)$	Initial cruise altitude
$c_3(x)$	Aircraft reference speed V_{ref}
$c_4(x)$	Excess fuel weight
$c_5(x)$ and $c_6(x)$	Wing flight controls actuation height clearance
$c_7(x)$ and $c_8(x)$	Wing flight controls actuation chord clearance
$c_9(x)$	Wing chord clearance for landing gear integration
$c_{10}(x)$	Wing tip chord
$c_{11}(x)$	Aircraft climb performance
$c_{12}(x)$	Aircraft mission range

The extended design structure matrix (XDSM) [22] of the two problems considered in Section 2.1 and 2.2 is presented in Figure 2. The MDO environment uses the optimization framework described in Section 3 that interfaces with an aircraft multidisciplinary analysis (MDA) environment. This MDA environment is comprised of sizing and simulation models of all major disciplines in aircraft conceptual design. The MDA uses design variables defined by Table 1 to first size aircraft engines (using a reference engine), wings, and structures (based on a reference structure) and performs low-speed and high-speed aerodynamics analyses. Then an aircraft balancing MDA loop is performed by assessing the mission performance of the sized aircraft based on fuel volume calculations, fuel burn curves, weight estimation of all aircraft components, tail sizing, and center of gravity envelopes. The balanced aircraft is then used to perform constraint checks in landing gear and flight control systems volumetric codes. The results of the MDA are fed back to the optimizer that collects all MDA parameters and quantities of

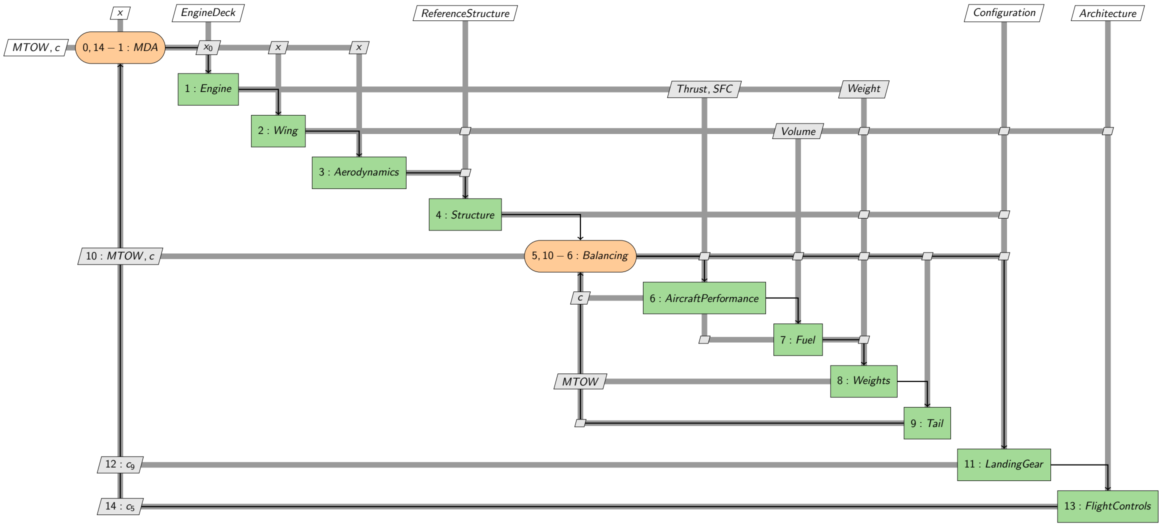


Figure 2: XDSM representation of the aircraft conceptual design optimization problem.

interest and associated design variables are obtained. The optimization process is performed by firstly defining a reference aircraft as a starting point and used later on a reference for engine and structures scaling and secondly defining the bounds of the design variables to meet aircraft design requirements which are defined as part of the constraints. It is noted that in this industrial MDO environment, any improvements of the minimum relative to the starting aircraft are deemed beneficial, for example, a 15% reduction in MTOW between the starting aircraft and the obtained minimum after optimization is considered a significant improvement. Due to Bombardier intellectual property considerations, some industrial cases have been modified and reproduced for the purpose of this work.

In the following subsections, we present descriptions of the two aircraft design problems subject to hidden constraints that will be used in this paper as the industrial applications.

2.1 Landing gear sizing simulation

For the first problem, a landing gear code from [38] was added to simulate a hidden constraint which is comprised of the following subroutines:

- Positioning subroutine: ground contact point positioning which starts with a defined wing and fuselage configuration. Given aircraft center of gravity (CG) limits and wing and fuselage geometry, the main and nose landing gear are positioned to satisfy a set of predefined constraints. Examples of such constraints are the tip over and tail strike angles shown in Figure 3.
- Sizing subroutine: structural sizing using three major load cases that typically size landing gear structure and then select main and nose landing gear tires and rims.
- Kinematics subroutine: kinematics of retraction analysis is performed to determine the stowage location of the gear and the feasibility of the proposed design from the previous processes.

Landing gear simulation failure in this problem is due to the kinematics subroutine and is driven by two distinct possible simulation failures. The first possible simulation failure occurs when calculating the extended position of a trailing arm type main landing gear as shown in Figure 4 (a). Compressed position is based on landing gear geometry, aircraft loads, wing or fuselage attachment points. The calculation of the extended position is based on aircraft weight, landing sink speed, compressed position, and fixed attachment points to the aircraft structure represented by the so-called pintle pin. In certain scenarios, the calculation of this extended position leads to an infeasible configuration and a simulation crash occurs. The second possible simulation failure occurs when analyzing landing gear kinematics

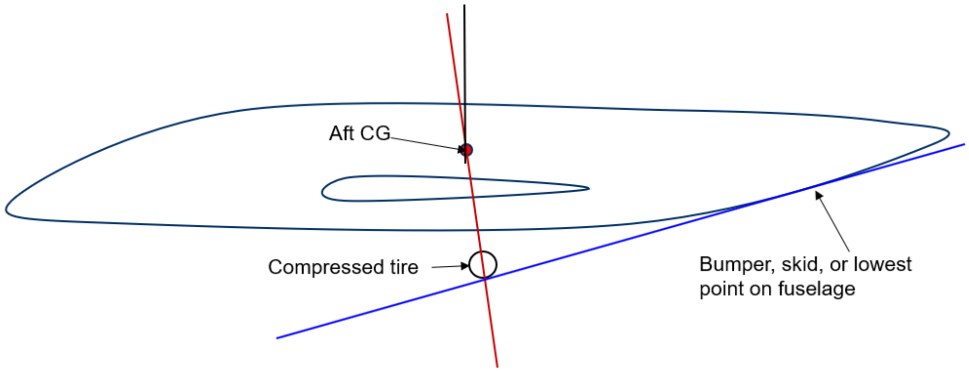
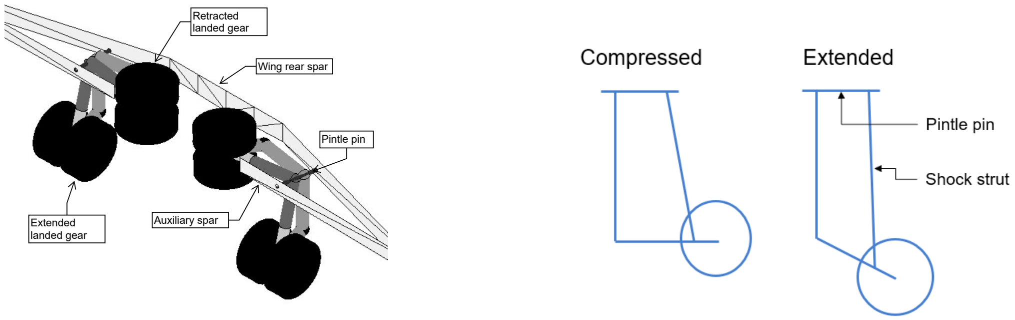


Figure 3: Landing gear positioning constraints examples showing tip over angle and tail strike angle (adapted from [10]).



(a) A landing gear retraction kinematics [38]

(b) A landing gear arrangement [36]

Figure 4: Representation of kinematics of retraction, compression, and extension of a trailing arm landing gear model

of retraction, e.g., simulation failure occurs if the position of the wing is not aligned with the landing gear bays which causes an error when calculating landing gear pintle pin (retraction axis) orientation. Figure 4 (b) shows an example aircraft illustrating a compressed landing gear and a corresponding retracted landing gear inside dedicated bays. We treat this code as a black box simulation. Therefore, we assume that we cannot adjust the internal code to avoid the hidden constraint or create a known constraint.

2.2 Aircraft performance simulation

A second problem that occurs due to an aircraft performance model simulation failure is presented herein. The aircraft performance model is responsible for analysis and simulation of aircraft ground, takeoff, climb, cruise, and descent performance. This model crashes when aircraft design variables lead to insufficient engine thrust at the beginning of the cruise flight phase, known as the initial cruise altitude. Takeoff is the first phase of flight starting from an initial aircraft velocity of zero and ending when the aircraft reaches an altitude of 35 ft as illustrated in Figure 5. An aircraft then transitions to a so-called *en route* climb where a rate of climb is set at a fixed aircraft speed until engine thrust is incapable to maintain the rate of a minimum set rate of climb. The initial cruise altitude is set when the minimum rate of climb condition is no longer met, and the engine thrust setting is adjusted to the cruise setting to meet the required cruise speed. Aircraft takeoff performance is driven by aircraft weight, drag, lift, engine thrust, airport altitude, and ground rolling friction. Climb and cruise performance are dependant on aircraft speed and altitude. The thrust produced by an aircraft engine is reduced with increasing speed of the aircraft and with increasing altitude. In this optimization

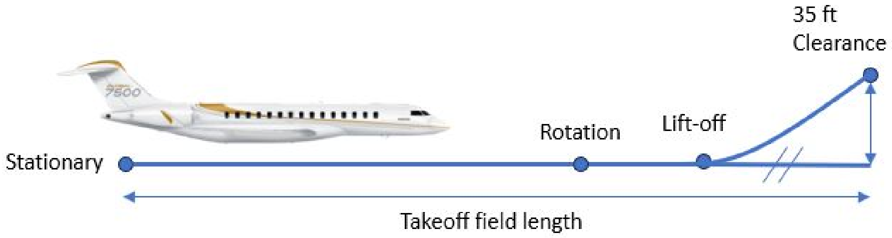


Figure 5: Aircraft takeoff phase illustration (adapted from [3]).

problem, design variables affecting wing design $x_2, \ldots, x_{12}$ impact drag and lift whereas the engine scaling factor x_1 impacts all thrust ratings and altitude/speed combinations. Failure of the simulation occurs when the combination of design variables lead to insufficient thrust to overcome aircraft drag at high speed at initial cruise altitude. The aircraft model would be able to simulate takeoff and climb at climb speed. However, the mission fails when initial cruise altitude is reached at the high speed aircraft requirement. Aircraft thrust requirement is calculated based on aircraft drag using [3]

$$D = \frac{\rho v^2 C_{\mathrm{d}} S_{\mathrm{ref}}}{2}, \tag{2}$$

where D is total aircraft drag, C_{d} is the drag coefficient, ρ is the dynamic pressure, v is aircraft speed, and S_{ref} is the reference aircraft wing area. T is engine thrust, and maximum T is defined based on x_1, whereas C_{d} and S_{ref} are dependant on $x_2, \ldots, x_{12}$. If D is higher than the T values at different flight phases determined using the engine scaling factor x_1, the code leads to a simulation failure as the aircraft performance model is not able to maintain the required aircraft speed. The non-linear relationships between D, S_{ref}, and $x_2, \ldots, x_{12}$ and the corresponding non-linear behaviour of this failure region presents a valuable demonstration of methods used to handle hidden constraints. This type of simulation failure falls within category (2) of hidden constraints since the failure occurs due to the specific software implementation of the aircraft performance model. Similar to the landing gear model in Section 2.1, we assume that the aircraft performance model is a black box simulation and a known constraint cannot be created to prevent this simulation failure.

3 Bayesian optimization with hidden constraints

A black box constrained surrogate-based optimization creates surrogate models of an objective y(x) and equality and inequality constraints $c_1(x), \ldots, c_m(x)$ that are evaluated using black box simulations without knowledge of the internal model of these simulations. In this work, an inequality-constrained surrogate-based optimization problems is formulated as

$$\begin{cases} \underset{x \in \Omega}{\text{minimize}} & \hat{y}(x) \\ \text{subject to} & \hat{c}_i(x) \leq 0, \ i = 1, \ldots, m, \end{cases} \tag{3}$$

where $\hat{y}(x)$ is a surrogate model of objective function and $\hat{c}_1(x), \ldots, \hat{c}_m(x)$ are surrogate models of the constraints. In the case of Bayesian optimization, the surrogate models are Gaussian process, or variations thereof, to be able to estimate probability distributions. These models are reconstructed during the optimization process. Prior to the optimization, a fixed number of evaluations design of experiments (DOE) is typically conducted to create the initial surrogate models and probability distributions. The optimization loop starts after the DOE evaluations as described in Algorithm 1. A so-called sequential enrichment problem is solved at every optimization iteration. The sequential enrichment problem aims to maximize an acquisition function that uses surrogate model values along with the probability distributions to balance exploration and exploitation of the optimization process. The solution of the sequential enrichment problem then recommends a new location in the design space for the objective and constraint functions to be evaluated using black box simulations. The newly evaluated location is then added to the data sets of the optimization. The optimization continues until a certain convergence threshold is met or until a maximum number of iterations is reached as per Algorithm 1.

Algorithm 1: The constrained Bayesian optimization framework.

```
Input: Initial DOE
Output: Best feasible point from the DOE
for $i = 1, \dots, max\_iter$ do
    Build GP surrogate models of the objective and constraints functions.;
    Maximize an acquisition function to find $x^{i+1}$. ;
    Evaluate objective and constraint functions at $x^{i+1}$.;
    Update the DOE.;
end
```

3.1 Feasibility enhanced acquisition functions

Several acquisition functions have been proposed and investigated, ranging from traditional and widely used functions such as probability of improvement (PI) [20] and expected improvement (EI) [19] to newly developed functions such as scaled Watson Barnes (WB2S) [6]. The probability of improvement [20] is given by

$$\mathrm{PI}(x) = \mathbb{P}(y(x) \leq y_{\min}) = \Phi\left(\frac{y_{\min} - \hat{y}(x)}{\hat{s}(x)}\right), \tag{4}$$

where $y_{\min}$ is the minimum value of the objective function observed so far, $\hat{y}(x)$ and $\hat{s}(x)$ are mean and standard deviation of the Gaussian process, and Φ is the normal cumulative distribution function.

The expected improvement (EI) [19] is of the form:

$$\mathrm{EI}(x) = (y_{\min} - \hat{y}(x))\,\Phi\left(\frac{y_{\min} - \hat{y}(x)}{\hat{s}(x)}\right) + \hat{s}(x)\,\phi\left(\frac{y_{\min} - \hat{y}(x)}{\hat{s}(x)}\right), \tag{5}$$

where $y_{\min}$ is the value of the incumbent objective function, $\hat{y}(x)$ and $\hat{s}(x)$ are mean and standard deviation of the Gaussian process. Φ and ϕ are the cumulative distribution function and probability density function of the Gaussian process respectively. If $\hat{s}(x) = 0$, $\mathrm{EI}(x)$ is set to zero.

The scaled Watson Barnes WB2S [6] is:

$$\mathrm{WB2S}(x) = s\mathrm{EI}(x) - \hat{y}(x), \tag{6}$$

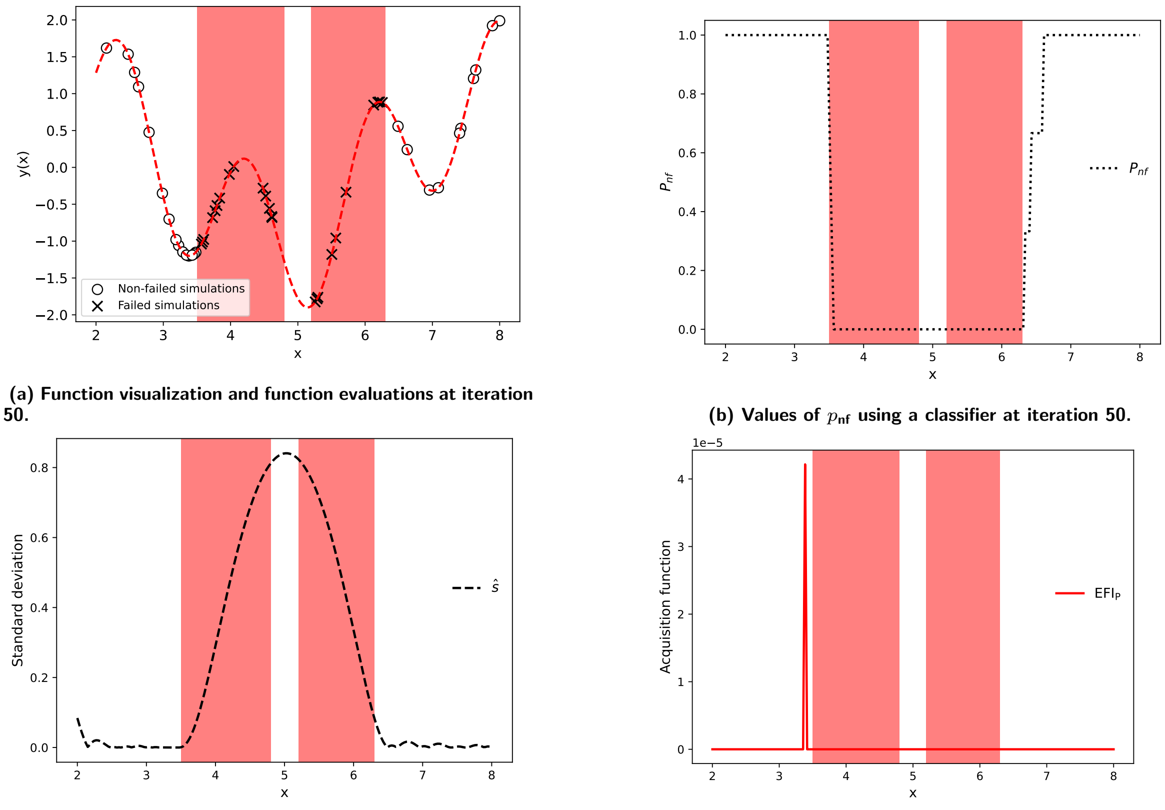
where s is a non-negative scaling factor defined in [6]. In previous work, methods to adapt BO to handle hidden constraints included the conditioning of the acquisition function by either the probability of non-failure or class of non failure [5, 15, 24, 33, 41]. Another method that is also used in the literature is constraining the design space of an optimization based on regions of predicted failures as shown in [2, 7, 33]. Typical expected feasible improvement acquisition functions are defined by:

$$\mathrm{EFI}_{\mathrm{P}}(x) = p_{\mathrm{nf}}(x)\;\mathrm{EI}(x) \tag{7}$$

and

$$\mathrm{EFI}_{\mathrm{C}}(x) = c_{\mathrm{nf}}(x)\;\mathrm{EI}(x), \tag{8}$$

where p_{nf} is the probability of non-failure and c_{nf} is the class of non-failure calculated using a surrogate model $Z(x)$. The main drawback of the existing methods that condition the acquisition function by the probability, or class, of non-failure is that during the early phase of an optimization process $\mathrm{EFI}_{\mathbf{P}}(x)$ and $\mathrm{EFI}_{\mathbf{C}}(x)$ could be incorrectly driven by the non-failure predictor, $Z(x)$, away from exploration regions of the design space especially if a non-sequential based framework is used. To demonstrate this behaviour, we present in Figure 6 a 1-dimensional function $y(x)$ constrained by two hidden simulation failure regions. The minimum of $y(x)$ lies in between these two hidden simulation failure regions. We use a BO algorithm per Algorithm 1 and a k-nearest neighbors classifier (kNN) at k=3 to calculate p_{nf}. After 50 iterations, the acquisition function no longer explores any of the hidden constraint regions nor the feasible region in between. This is driven by the non-failure predictor, $Z(x)$, where $\mathrm{EFI}_{\mathbf{P}}(x)$ is estimated to be equal to zero even within the feasible region where the minimum lies. In addition, global convergence cannot always be guaranteed using the expected feasible improvement acquisition functions in Eq. (7) and Eq. (8) for all types of classifiers. Global convergence proof of Eq. (7) has been shown using a Gaussian process classifier in [5], however using different types of classifiers do not always guarantee global convergence. For a BO algorithm that uses a Gaussian process $\zeta(x)$, an acquisition function can be written in a form that separates an exploitation portion from an exploration portion as presented in the formulation of EI in Eq. (5). The exploration term is dependant on the standard deviation term $\hat{s}(x)$. We hypothesize that in regions of the design space where $\hat{s}(x)$ is high for $\zeta(x)$, the non-failure predictor model $Z(x)$ would be inaccurate. In order to minimize the impact of the inaccuracy of Z on the acquisition function, we try to reduce the influence of the former on the exploration region of the latter by means of an exploration factor α. Therefore, for a given $x \in \Omega$ and as far as $\hat{s}(x) \neq 0$, a feasibility enhanced expected improvement acquisition function $\mathrm{EFI}_{\mathrm{FE}}(x)$ at x is



(a) Function visualization and function evaluations at iteration 50.

(b) Values of p_{nf} using a classifier at iteration 50.

(c) Values of $\hat{s}(x)$ at iteration 50.

(d) Values of the $\mathrm{EFI}_{\mathbf{P}}(x)$ at iteration 50.

Figure 6: 1-dimensional function BO example ($\sin(x) + \sin(\frac{10}{3}x)$)using $\mathrm{EFI}_{\mathbf{P}}(x)$ showing the hidden regions ($3.5 < x < 4.8$ and $5.2 < x < 6.3$) (in red).

defined by:

$$\mathrm{EFI}_{\mathrm{FE}}(x) = p_{\mathrm{nf}}(x)\left(y_{\min} - \hat{y}(x)\right)\Phi\left(\frac{y_{\min} - \hat{y}(x)}{\hat{s}(x)}\right) + p_{\mathrm{nf}}(x)^{\alpha(x)}\hat{s}(x)\phi\left(\frac{y_{\min} - \hat{y}(x)}{\hat{s}(x)}\right). \tag{9}$$

If $\hat{s}(x) = 0$, $\mathrm{EFI}_{\mathrm{FE}}(x)$ is set to zero. The parameter $\alpha \in [0,1]$ can be seen as an exploration factor that allows the acquisition function to approach failure regions of the design space.

The $\mathrm{EFI}_{\mathrm{FE}}$ acquisition function conditions the exploitation portion of EI similar to $\mathrm{EFI}_{\mathrm{P}}$; however the impact on the exploration portion is minimized by the term α. We note that when $\alpha = 1$ one has $\mathrm{EFI}_{\mathrm{FE}} = \mathrm{EFI}_{\mathrm{P}}$. We propose a dynamic calculation of the term α during an optimization based on x. The goal of this dynamic calculation is to prevent the acquisition function from solely relying on p_{nf} considering the example shown in Figure 6. Therefore, we use a term $\bar{s_c}(x)$ that relies on information from $\hat{s}(x)$ of ζ to determine the value of α as follows

$$\alpha(x) = \begin{cases} 1.0, & \text{if } \bar{s_c}(x) \leq \epsilon, \\ 0.0, & \text{if } \bar{s_c}(x) > \epsilon \text{ and } p_{\mathrm{nf}}(x) = 0, \\ \alpha_0, & \text{if } \bar{s_c}(x) > \epsilon, \end{cases} \tag{10}$$

where ϵ is a pre-specified tolerance and $\bar{s_c}(x)$ is defined as follows

$$\bar{s_c}(x) = \begin{cases} 0.0, & \text{if } x \in A \\ \hat{s}(x), & \text{otherwise}, \end{cases} \tag{11}$$

where A is a set containing all simulation failure data with a predefined tolerance. The fixed term α_0 is the fixed exploration factor that allows the acquisition function to explore closer to a failure region even if p_{nf} is low. We also note that another approach instead of Eq. (11) could be to directly equate $\bar{s_c}(x) = \hat{s}(x)$ and apply a filter after the acquisition function optimization results to remove any values that lie within A. To visualize the impact of $\alpha(x)$ on the acquisition function, we consider the same example from Figure 6 and calculate $\mathrm{EFI}_{\mathrm{FE}}$ as shown in Figure 7. We note that the exploration portion of the acquisition function is not affected by p_{nf} as opposed to $\mathrm{EFI}_{\mathrm{P}}$ from Figure 6 (d). In this case, the impact of α_0 is negligible since $\alpha(x)$ falls either under the first or the second conditions from Eq. (10). Nonetheless, it is evident that the acquisition function in this case will keep exploring the regions where $p_{\mathrm{nf}} = 0$ until the design space meets the standard deviation tolerance ϵ.

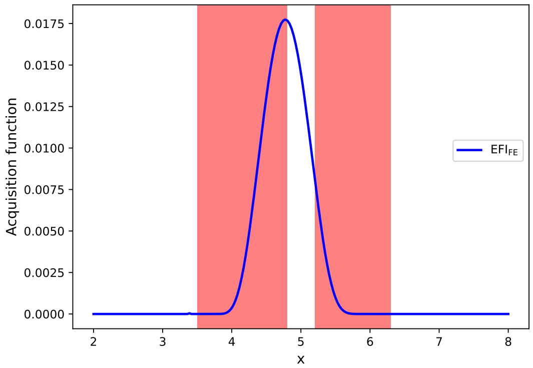


Figure 7: $\mathrm{EFI}_{\mathrm{FE}}$ values of the 1-dimensional function after the same 50 function evaluations from Figure 6.

In this paper, we implement and test the proposed acquisition function based on EI, however the approach can be expanded for any type of acquisition function where its formulation allows the separation between exploration and exploitation. For example, the WB2S acquisition function in Eq. (6) can be reformulated as

$$\mathrm{WB2S}_{\mathrm{FE}}(x) = s\mathrm{EFI}_{\mathrm{FE}}(x) - p_{\mathrm{nf}}(x)\hat{y}(x). \tag{12}$$

A Bayesian optimization algorithm can be adapted to handle hidden constraints using $\mathrm{EFI}_{\mathrm{FE}}(x)$ as shown in Algorithm 2. It is noted in this updated algorithm that the classifier model of the hidden constraint is updated once at every loop of the Bayesian optimization and that the probability of feasibility is then calculated using this classifier model at every call within the sequential enrichment problem to calculate $\mathrm{EFI}_{\mathrm{FE}}(x)$. This approach reduces the impact of building the classifier model on computational costs by building a single classifier model for all hidden constraints. Calculation of p_{nf} is dependant on the selection of the failure classifier Z and is shown in Section 3.2.

Algorithm 2: Bayesian optimization with hidden constraints.

```
Input: Initial DOE and simulation failure set $A$
Output: Best feasible point from the DOE
for $i = 1, \dots, max\_iter$ do
    Build GP surrogate models of the objective and constraints functions.;
    Build failure classifier $Z$ to estimate the probability of non-failure $p_{\mathrm{nf}}$.;
    Estimate parameter $\alpha$ using Eq. (10).;
    Maximize a $p_{\mathrm{nf}}$ based acquisition function ($\mathrm{EFI}_{\mathrm{FE}}$ or $\mathrm{WB2S}_{\mathrm{FE}}$) to find $x^{i+1}$. ;
    Evaluate objective and constraint functions at $x^{i+1}$.;
    Update the DOE and simulation failure set $A$.;
end
```

3.2 Non-failure probability estimation

Several models have been proposed in the literature to represent hidden constraints using supervised ML techniques. The use of Gaussian process classifiers, conditioned by signs of observations was proposed in [5]. Random forests were used in [24]. We consider additional popular ML classification means to compare with existing literature, including nearest neighbor classifiers, decision trees and rule-based classifiers, probabilistic models, and support vector machines. We use labeled data from failed evaluations to construct and adapt these classifiers on supervised data.

The k-nearest neighbors classifier is commonly based on the Euclidean distance d between a sample x and the specified training samples $x_{train} \in N$ samples [1]. In a 2 class set where simulation failure is one class and non-failure is another, the probability of the non failure class at a given point x is computed by

$$p_{\mathrm{nf}_{\mathrm{kNN}}}(x) = \frac{\sum_{i \in N} I(x) d(x, x_i)}{\sum_{i \in N} d(x, x_i)}, \tag{13}$$

where $d(x, x_i)$ is the distance between the point x and a training point x_i and $I(x)$ is an index function that is equal to one when the predicted class is non-failure and zero otherwise [27].

Decision trees use a set of tree-like hierarchical decisions on the input variables to model the classification process; however, such methods may suffer from over-fitting or coarse approximations of a true classification boundary layer if the amount of training data is insufficient [16]. In this context, the probability for a decision tree for p_{nf} is computed at a terminal node m of the tree with N_m samples by

$$p_{\mathrm{nf}_{\mathrm{DT}}}(x) = \frac{1}{N_m} \sum_{x \in N_m} I(x). \tag{14}$$

Probabilistic classifiers such as logistic regression construct a relationship between the input features and output class as a probability [1]. In logistic regression, the probability of a class-membership is expressed in terms of feature variables using a discriminative function. In a binary classification problem of failure and non-failure, the probability of an instance x belonging to the non-failure class is modeled using the logistic function from [1, 16]

$$p_{\mathrm{nf}_{\mathrm{LR}}}(x) = \frac{1}{1 + \exp\left(\theta_0 + \theta^{\top} x\right)}, \tag{15}$$

where θ_0 is an offset parameter and θ is a coefficient with the same dimensions as x. Training a logistic regression model entails solving an optimization problem that maximizes a likelihood function using (θ_0, θ) as design variables where the likelihood function is defined as the product of the probabilities of all the training examples predicting their assigned classes using Eq. (15).

Non-linear support vector machines (SVM) classify instances by defining a boundary that separates classes of samples from a dataset. SVM does not directly compute probability to obtain class predictions. We present the method used in [27] to compute this probability. In a case of two class classification, the class probabilities are calibrated using the scaling proposed in [29].

$$p_{\mathrm{nf}_{\mathrm{SVM}}}(x) = \frac{1}{1+\exp\left(af(x)+b\right)}, \tag{16}$$

where $f(x)$ is the SVM uncalibrated prediction of the hidden constraint at x and parameters a and b are found by minimizing an error function on the training data [29]. The uncalibrated SVM model prediction $f(x)$ is obtained using data from N training samples as follows

$$f(x) = \sum_{i=1}^{N} \lambda_i z_i k(x_i, x),$$

where x_i is the ith training sample, λ_i is the corresponding Lagrangian multiplier of the sample and is obtained by solving a so-called Lagrangian relaxation problem defining the boundary of the SVM, z_i is the class of the training sample, and k is a kernel function used to handle non-linearity in the defined boundary. In this paper, we use the Gaussian radial basis function kernel for SVM modeling similar to [2].

Gaussian processes are also used as classifiers to estimate the probability of non-failure [42].

[5] proposed a Gaussian process classifier (GPC) method to modify the acquisition function of a BO as in Eq. (7) in order to handle hidden constraints. The proposed GPC is conditioned on the signs of the observations rather than their values where p_{nf} is approximated as follows after sampling N samples from the probability density function of the Gaussian process:

$$p_{\mathrm{nf}_{\mathrm{GPC}}}(x) = \frac{1}{N}\sum_{i\in N} \bar{\phi}\left(\frac{-\hat{c}(x)}{\hat{s}_c^2(x)}\right), \tag{17}$$

where N is the number of observation samples, and $\hat{c}(x)$ and $\hat{s}_c^2(x)$ are the mean and variance of the Gaussian process of the constraint at x. $\bar{\phi}$ is calculated from the standard Gaussian cumulative distribution function:

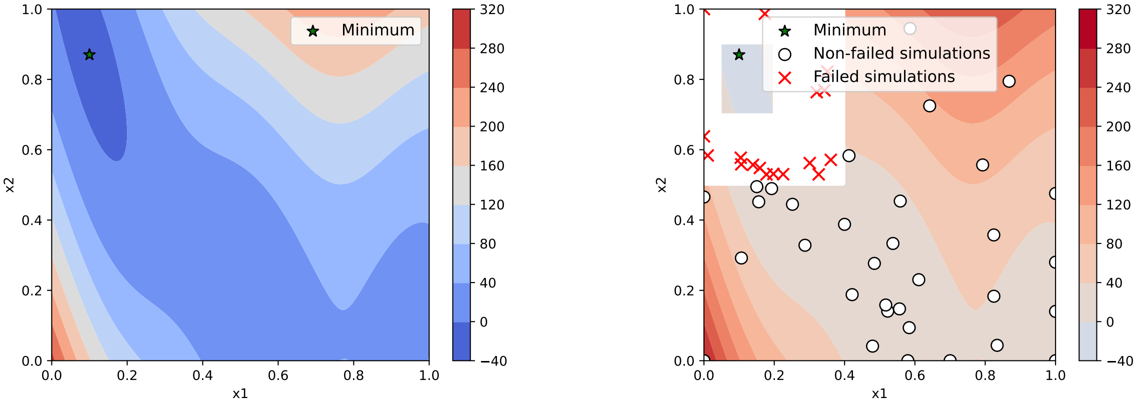
$$\bar{\phi}(\frac{a}{b}) = \begin{cases} 1-\phi(\frac{a}{b}), & \text{if } b \neq 0 \\ 1, & \text{if } b = 0. \end{cases}$$

3.3 Illustration example

The behaviour of the proposed method (see Algorithm 2) is illustrated using an analytical example showing the impact on the acquisition function and convergence. $\mathrm{EFI}_{\mathrm{P}}$ and $\mathrm{EFI}_{\mathrm{FE}}$ functions and selected classifiers are implemented in an open-source Python Bayesian optimization tool [26] in addition to Scikit-learn libraries [27].The selected illustration example is an unconstrained optimization problem based on the scaled Branin-Hoo function presented in [12] shown by Eq. (18) and in Figure 8 (a).

$$\begin{cases} \underset{x_1,x_2\in\mathbb{R}^2}{\text{minimize}}\ f(x_1,x_2) \\ f(x_1,x_2) = \left(\bar{x_2} - \dfrac{5.1\bar{x_1}}{4\pi^2} + \dfrac{5\bar{x_1}}{\pi} - 6\right)^2 + 10\left((1-\dfrac{1}{8\pi})\cos\bar{x_1} + 1\right) + 3\bar{x_1} \\ \text{where, } \bar{x_1} = 15x_1 - 5, \bar{x_2} = 15x_2 \\ \text{and } x_1, x_2 \in [0,1] \end{cases} \tag{18}$$

A hidden simulation failure region H is defined where $f(x_1, x_2)$ returns non-valid values (i.e., Nan) when $x_1 < 0.4$ and $x_2 > 0.5$ except for an inner region where the minimum lies where $0.05 < x_1 < 0.2$ and $0.7 < x_2 < 0.9$. A 50 evaluation optimization using EFI_P is performed to test the behaviour of the acquisition function. The hidden region H is illustrated in Figure 9 (a) and the updated function including H is shown in Figure 8 (b). It is noted that H is selected so that the minimum of the unconstrained problem shown in Figure 8 (a) lies in the feasible region within H in order to highlight the impact of EFI_FE acquisition function when the minimum is close to a failure region assuming that p_nf would be inaccurate in such a region.

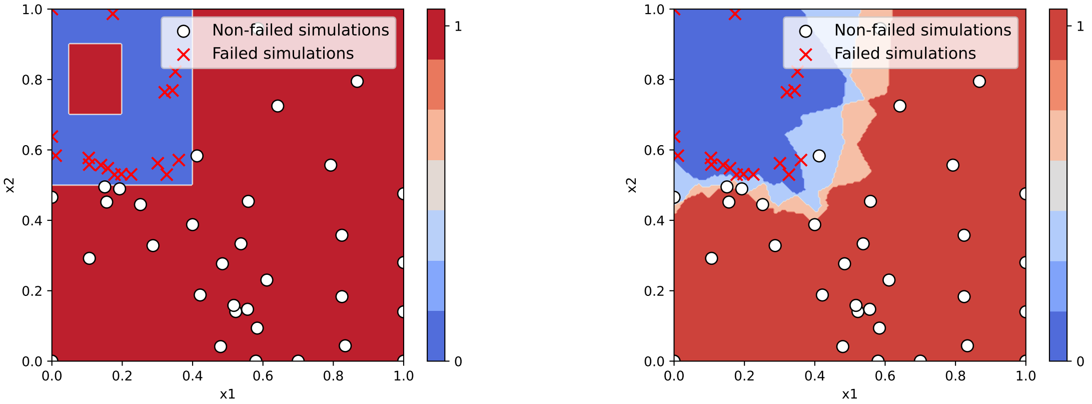


(a) Branin-Hoo objective function

(b) Masked Branin-Hoo function

Figure 8: Visualization of the unmasked (contour map) and masked (white area) Branin-Hoo function showing results of a 50-evaluation optimization using EFI_P.

The 50 evaluation points are used to create a Gaussian process of the objective function and a classifier of the hidden region to predict the probability of non-failure p_nf. The classifier selected is a kNN classifier with $k = 3$ and p_nf calculations using the classifier are shown as contour plots in Figure 9 (b).

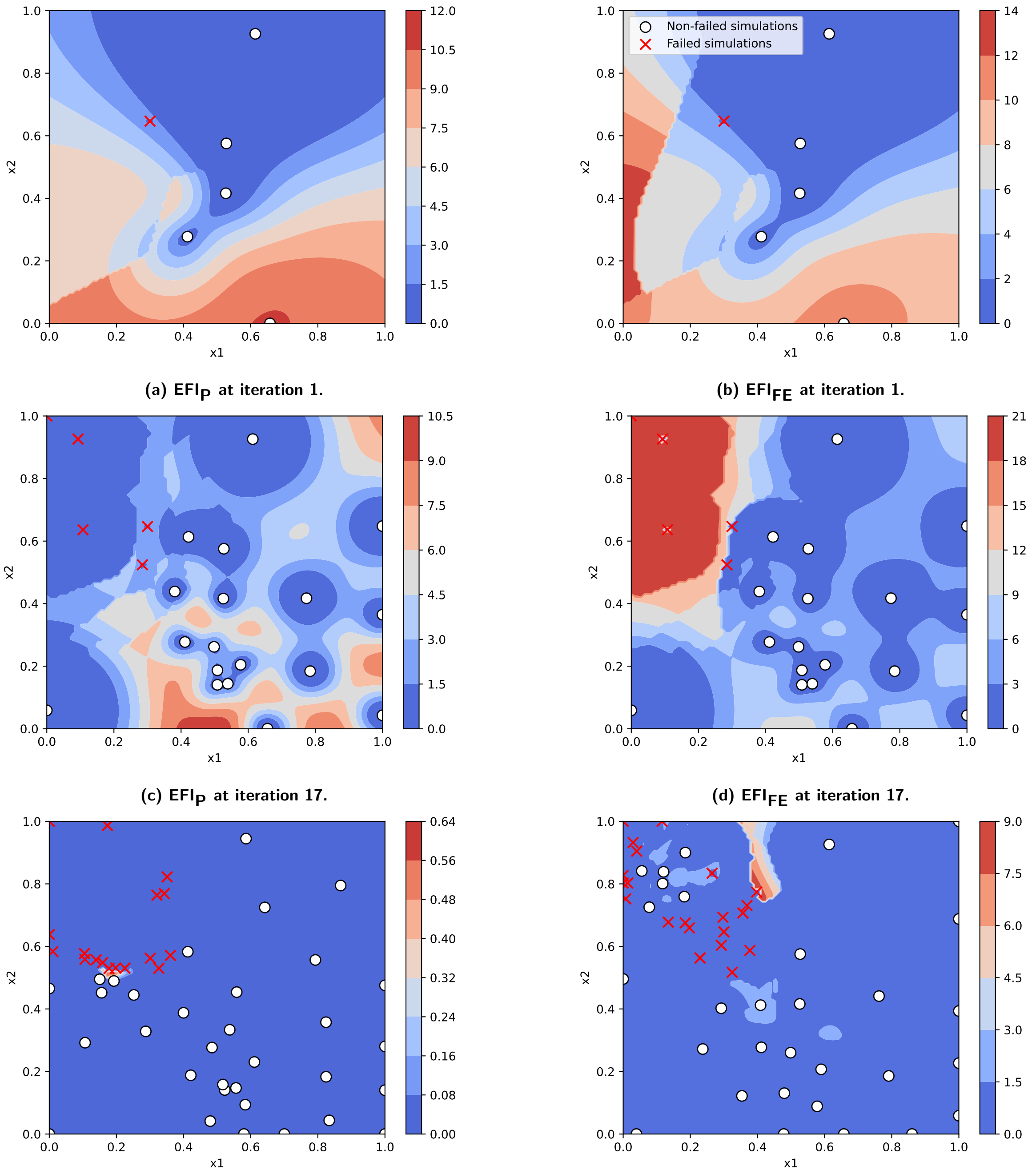


(a) Hidden region (in blue) and the non-failure region (in red) at iteration 50.

(b) Values of p_nf using a kNN classifier with $k = 3$ at iteration 50.

Figure 9: Hidden region and classifier visualization using the 50-iteration DOE.

Using the kNN classifier with $k = 3$, EFI_FE acquisition function is compared against EFI_P using the same classifier in Figure 10 where red contours represent higher values and blue contours present lower values. Firstly, comparing the two acquisition function at the first iteration in Figure 10 (a) and (b), we note that the behaviour of EFI_FE differs in the left side of the Figure which can be



(a) $\mathbf{EFI_P}$ **at iteration 1.**

(b) $\mathbf{EFI_{FE}}$ **at iteration 1.**

(c) $\mathbf{EFI_P}$ **at iteration 17.**

(d) $\mathbf{EFI_{FE}}$ **at iteration 17.**

(e) $\mathbf{EFI_P}$ **at iteration 50.**

(f) $\mathbf{EFI_{FE}}$ **at iteration 50.**

Figure 10: Acquisition function comparison at the first iteration, 17th iteration, and 50th iteration during an optimization starting with a DOE of 5 samples.

attributed to high variance and a prediction of p_{nf} of 0. We also note that $\mathrm{EFI}_{\mathrm{P}}$ favors exploitation outside the hidden constraint region H compared to $\mathrm{EFI}_{\mathrm{FE}}$. Secondly, we performed an optimization using $\mathrm{EFI}_{\mathrm{FE}}$ and we show the acquisition function at the iteration where the failure region starts to be explored (i.e., iteration 17 in Figure 10 (d)). Using the same 17 evaluations we show $\mathrm{EFI}_{\mathrm{P}}$ in Figure 10 (c) and we note that: 1) $\mathrm{EFI}_{\mathrm{P}}$ prevents the optimizer from selecting future evaluation points within H due to p_{nf}, and 2) $\mathrm{EFI}_{\mathrm{FE}}$ favors exploring regions where there is a high variance of the classifier $Z(x)$. Finally, at iteration 50 using two different acquisition functions in Figure 10 (a) and (b), we can see that using EFI_{FE} enables the optimizer to find the minimum in Figure 8 (b) whereas EFI_{P} still cannot access the region H.

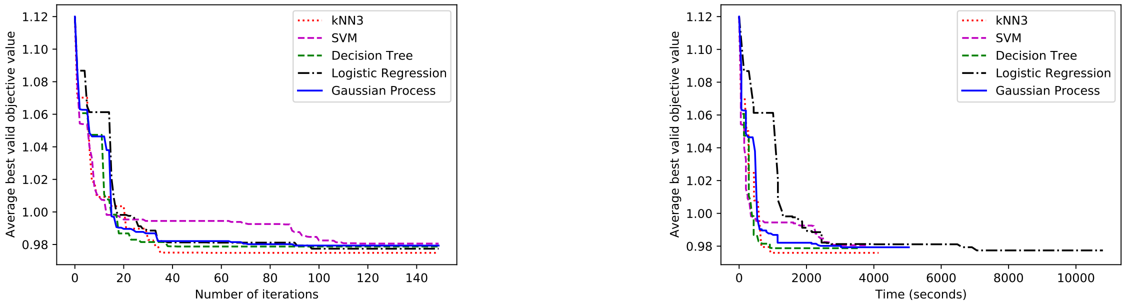
In Section 4, we use optimization results directly to assess the use of EFI_{FE} using the industrial application problems considered in this paper.

4 Aircraft design optimization

The results of the industrial application problems introduced in Section 2 are presented here. All results are obtained using an Intel® Xeon® CPU E5-1650 v3 @ 3.50 GHz core and 32 GB of memory. Optimization results are normalized with respect to the considered baseline aircraft design. We first compare results based on the choice of classifiers using the landing gear sizing problem from Section 2.1. The five tested classifiers, as detailed in Section 3.2, are:

1. a k-nearest neighbors classifier with k=3 (`kNN3`),
2. an SVM classifier (`SVM`),
3. a Gaussian process classifier (`Gaussian Process`),
4. a decision tree classifier (`Decision Tree`), and
5. a logistic regression classifier (`Logistic Regression`).

We conducted 10 optimization runs using a DOE with as many sample points equal to the number of design variables and 150 BO iterations. Figure 11 shows that the `kNN3` classifier yields the best convergence rate and lowest computational time. For that reason, in the remainder of this paper, a `kNN3` classifier will be used.

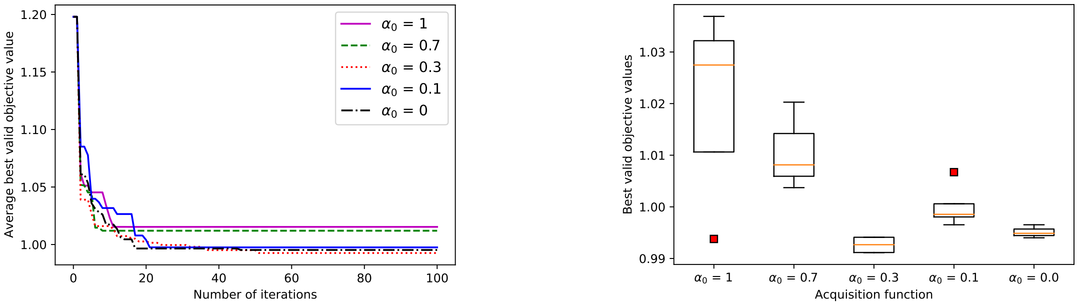


(a) **(b)**

Figure 11: Convergence of the landing gear sizing problem with respect to (a) number of iterations and (b) computational time for an average of 10 optimization runs.

The impact of the exploration factor α_0 on optimization convergence was also studied. Figure 12 depicts the convergence rate using 5 different values of α_0: 1, 0.7, 0.3, 0.1, and 0. We first note that at $\alpha_0 = 1$ and $\alpha_0 = 0.7$ the corresponding convergence rate is inferior compared to other lower values of α_0. This is expected since a lower value of α_0 leads to more acquisition function exploration in regions where p_{nf} is low, and if the minimum lies close to the simulation failure regions, p_{nf} may estimate low probability of simulation non-failure at the minimum. It is also noted that convergence plots of $\alpha_0 = 0.3, 0.1, 0$ are similar in minimum value and rate of convergence with $\alpha_0 = 0.3$ having the lowest minimum marginally. For that reason, in the remainder of this paper, an α_0 value of 0.3 will be used.

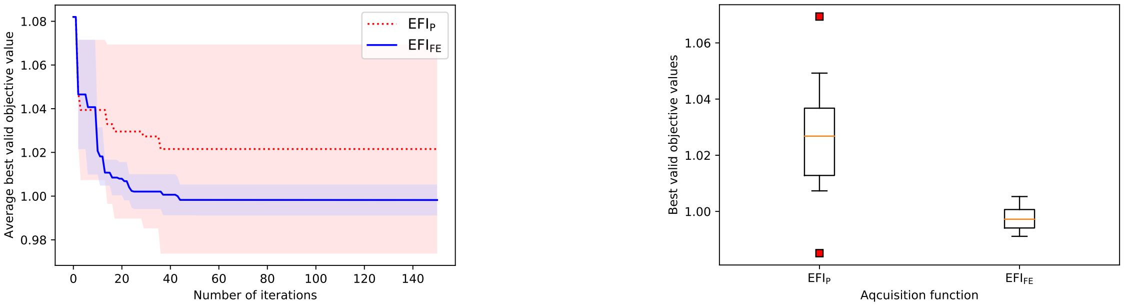
We used an average of 20 optimization runs for the two industrial problems, landing gear sizing and aircraft performance simulation, to perform the comparison between EFI_{FE} and EFI_{P}. Results for the landing gear sizing problem and aircraft performance simulation problem are presented in Figure 13 and Figure 14. Convergence plots show the average of the minimum normalized objective values at



(a) Convergence plots

(b) Box plots of the obtained solutions

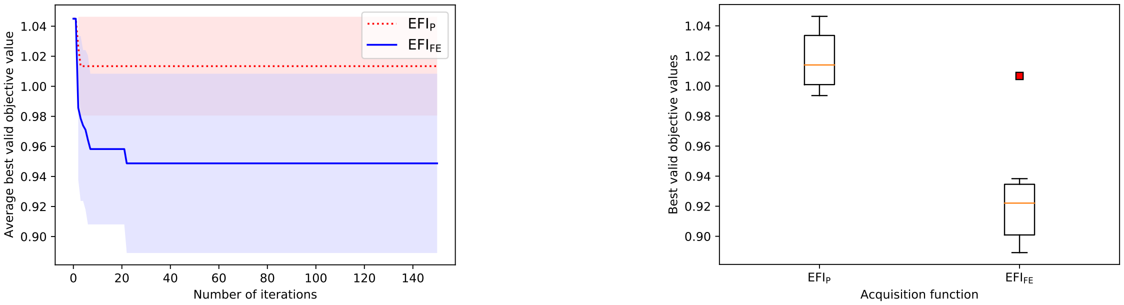
Figure 12: Comparison of landing gear sizing problem results (average of 10 runs) for different values of α_0.



(a) Convergence plots

(b) Box plots of the obtained solutions

Figure 13: Comparison of obtained results for the landing gear sizing problem using a kNN classifier with $k = 3$ (average of 20 optimization runs).



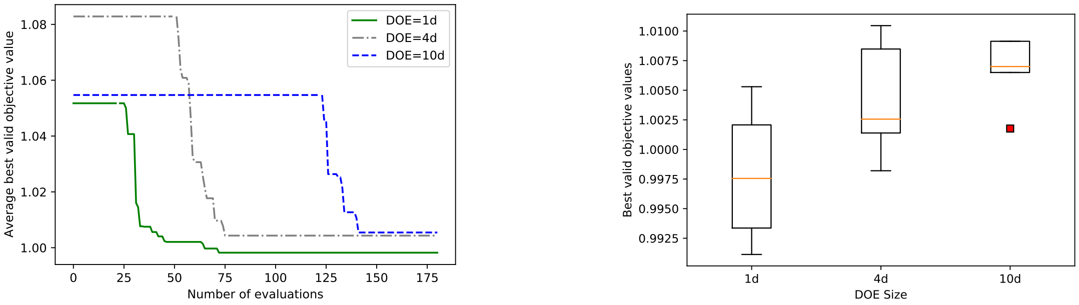
(a) Convergence plots

(b) Box plots of the obtained solutions

Figure 14: Comparison of obtained results for the aircraft performance problem using a kNN classifier wit $k = 3$ (average of 20 optimization runs).

every iteration (shown as a line per acquisition function) and the variance at every iteration (shown as pastel colors of the corresponding line).

The robustness of the proposed approach against the choice of the number of DOE evaluations is verified to check the sensitivity of optimization results to the choice of the size of the initial DOE. Figure 15 depicts the convergence rate using different initial values of DOE evaluations: $1d, 3d$, and $10d$, where d is the number of design variables of 12. Results are presented such that the DOE evaluations are shown as a flat horizontal line presenting the best valid objective value of the DOE until a better minimum is found. We note that DOE evaluations of $1d$ and $4d$ are able to converge to similar optimum values whereas the $10d$ DOE evaluation results in a slightly inferior convergence and a higher optimum. However, we conclude that the presented optimization framework and the acquisition function are robust with respect for the number of evaluations of the DOE, and that the suggested number of evaluations to be between $1d$ and $4d$.



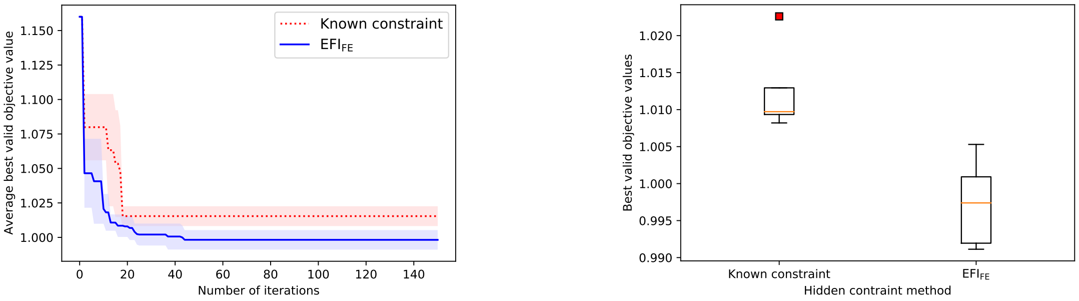
(a) Convergence plots

(b) Box plots of the obtained solutions

Figure 15: Comparison of obtained results for the landing gear sizing problem for different DOE sizes (average of 10 runs).

It is noted that in this work, the aircraft design problems along with the simulation failure problems have been chosen purposefully to highlight the behaviour of the proposed Bayesian algorithm with EFI_{FE}, that is by selecting a limited design space where a major part of which is within regions of the hidden constraints. As presented in Sections 2.1 and 2.2, the hidden failure regions are well understood. However, for similar problems where the design space or the hidden failure regions are within a black box simulation model invisible to the optimization engineer, we recommend to perform a thorough analysis of the design space to avoid the presence of any hidden constraints unrelated to the black box simulation model as discussed in Section 2. Another approach would be to analyze the design space and create a known constraint of the hidden failure region. We tested such an approach on the landing gear sizing problem from Section 2.1 by creating known constraints of the two possible simulation failure scenarios.

We also compared EFI_{FE} to a scenario where the simulation failure is a known constraint. We used the landing gear simulation in Section 2.1 and we adjusted the code to return a value to be used as the known constraint instead of what would have been a simulation failure. This scenario is normally not feasible if the simulation causing failures is a black box simulation that cannot be adjusted. We only consider this scenario here to compare the behaviour of EFI_{FE} when compared to a conventional BO with known constraint where the known constraints are modeled using Gaussian processes. Figure 16 compares the convergence between the known constraints method (labeled as `Known constraint`) and



(a) Convergence plots

(b) Box plots of the obtained solutions

Figure 16: Comparison of obtained results between using a known constraint Gaussian process and a hidden constraint with a `kNN3` classifier (average of 10 runs).

EFI_{FE} using a `kNN3` classifier using an average of 10 different initial DOE runs since the number of constraints is different due to the addition of the known constraint. It is noted that both approaches lead to similar convergence rates with the hidden constraint using a `kNN3` classifier producing a slightly lower minimum. This can be attributed to the fact that the constrained failure approach uses a Gaussian process with radial bases function (RBF) kernel to model the constraints (as is the standard approach used in the Bayesian optimizer from [26]). A Gaussian process with a RBF kernel models the boundary of the constraint as a smooth transition. However, we know that the failure regions presented in the landing gear sizing problem possess a sharp boundary between the non-failure simulation region and the failure simulation region, and a kNN3 classifier is better suited for such deterministic failures.

5 Conclusion

We investigated solutions to handle hidden constraints in aircraft design optimization problems. We targeted black box simulation models with failure as the candidates of hidden constraints. Then we used a feasibility enhanced acquisition function, $\mathrm{EFI}_{\mathrm{FE}}$, in a Bayesian optimization algorithm to perform aircraft conceptual design optimizations. Finally, we validated $\mathrm{EFI}_{\mathrm{FE}}$ using two industrial aircraft conceptual design problems based on landing gear sizing and kinematic simulation and aircraft performance simulation. Using the two industrial problems, we performed comparative analyses relative to the choice of the supervised ML classifiers used in addition to the internal exploration factor α of $\mathrm{EFI}_{\mathrm{FE}}$. We also showed the benefits of $\mathrm{EFI}_{\mathrm{FE}}$ over existing methods in literature with respect to convergence rates and optimum values. Future work on this topic includes extending the application to more industrial test cases with varying fidelity of the simulation models to understand the impact of the choice of simulation fidelity on optimizations with hidden constraints.

References

[1] Charu C Aggarwal. Data mining: the textbook. Springer, Yorktown Heights, 2015.

[2] Candelieri Antonio. Sequential model based optimization of partially defined functions under unknown constraints. Journal of Global Optimization, 79:1–23, 2019.

[3] Mario Asselin. An introduction to aircraft performance. AIAA, Reston, VA, 1997.

[4] Charles Audet, Gilles Caporossi, and Stéphane Jacquet. Binary, unrelaxable and hidden constraints in blackbox optimization. Operations Research Letters, 48:467–471, 2020.

[5] François Bachoc, Céline Helbert, and Victor Picheny. Gaussian process optimization with failures: classification and convergence proof. Journal of Global Optimization, 78:483–506, 2020.

[6] Nathalie Bartoli, Thierry Lefebvre, Sylvain Dubreuil, Romain Olivanti, Rémy Priem, Nicolas Bons, Joaquim RRA Martins, and Joseph Morlier. Adaptive modeling strategy for constrained global optimization with application to aerodynamic wing design. Aerospace Science and technology, 90:85–102, 2019.

[7] Anirban Basudhar, Christoph Dribusch, Sylvain Lacaze, and Samy Missoum. Constrained efficient global optimization with support vector machines. Structural and Multidisciplinary Optimization, 46:201–221, 2012.

[8] Jasper H Bussemaker, Thibault De Smedt, Gianfranco La Rocca, Pier Davide Ciampa, and Björn Nagel. System architecture optimization: An open source multidisciplinary aircraft jet engine architecting problem. In AIAA Aviation 2021 Forum, page 3078, 2021.

[9] Richard Curran, Mark Price, Srinivasan Raghunathan, Emmanuel Benard, Stephen Crosby, Sylvie Castagne, and Paul Mawhinney. Integrating aircraft cost modeling into conceptual design. Concurrent Engineering, 13:321–330, 2005.

[10] Norman S Currey. Aircraft landing gear design: principles and practices. AIAA, Reston, VA, 1988.

[11] Paul Feliot, Yves Le Guennec, Julien Bect, and Emmanuel Vazquez. Design of a commercial aircraft environment control system using bayesian optimization techniques. 5th International Conference on Engineering Optimization, 2016.

[12] Alexander Forrester, Andras Sobester, and Andy Keane. Engineering design via surrogate modelling: a practical guide. John Wiley & Sons, New Jersey, NY, 2008.

[13] Mark Gammon. A review of common geometry issues affecting mesh generation. In 2018 AIAA Aerospace Sciences Meeting, page 1402, 2018.

[14] K. O. Geddes, S. R. Czapor, and G. Labahn. Algorithms for Computer Algebra. Kluwer, Boston, 1992.

[15] Michael A. Gelbart, Jasper Snoek, and Ryan P. Adams. Bayesian optimization with unknown constraints. In Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence, UAI'14, page 250–259, 2014.

[16] Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. The elements of statistical learning: data mining, inference, and prediction, volume 2. Springer, New York, NY, 2009.

[17] Ryan P Henderson, Joaquim RRA Martins, and Ruben E Perez. Aircraft conceptual design for optimal environmental performance. The Aeronautical Journal, 116:1–22, 2012.

[18] Timothy MS Jim, Ghifari A Faza, Pramudita S Palar, and Koji Shimoyama. Bayesian optimization of a low-boom supersonic wing planform. AIAA journal, 59:4514–4529, 2021.

[19] Donald R Jones, Matthias Schonlau, and William J Welch. Efficient global optimization of expensive black-box functions. Journal of Global optimization, 13:455–492, 1998.

[20] Harold J Kushner. A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. 1964.

[21] Rémi Lam, Matthias Poloczek, Peter Frazier, and Karen E Willcox. Advances in Bayesian optimization with applications in aerospace engineering. In 2018 AIAA Non-Deterministic Approaches Conference, page 1656, 2018.

[22] Andrew B Lambe and Joaquim RRA Martins. Extensions to the design structure matrix for the description of multidisciplinary design, analysis, and optimization processes. Structural and Multidisciplinary Optimization, 46:273–284, 2012.

[23] Sébastien Le Digabel and Stefan M Wild. A taxonomy of constraints in black-box simulation-based optimization. Optimization and Engineering, 2023.

[24] H Lee, R Gramacy, Crystal Linkletter, and G Gray. Optimization subject to hidden constraints via statistical emulation. Pacific Journal of Optimization, 7:467–478, 2011.

[25] J.R.R.A. Martins. Aerodynamic design optimization: Challenges and perspectives. Computers & Fluids, 239:105391, 2022.

[26] Fernando Nogueira. Bayesian Optimization: Open source constrained global optimization tool for Python, 2014.

[27] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12:2825–2830, 2011.

[28] P Piperni, A DeBlois, and R Henderson. Development of a multilevel multidisciplinary-optimization capability for an industrial environment. AIAA journal, 51:2335–2352, 2013.

[29] John Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. Advances in large margin classifiers, 10:61–74, 1999.

[30] Remy Priem, Hugo Gagnon, Ian Chittick, Stephane Dufresne, Youssef Diouane, and Nathalie Bartoli. An efficient application of Bayesian optimization to an industrial MDO framework for aircraft design. AIAA Aviation Forum, 2020.

[31] Thomas A Reist, David Koo, David W Zingg, Pascal Bochud, Patrice Castonguay, and David Leblond. Cross-validation of high-fidelity aerodynamic shape optimization methodologies for aircraft wing-body optimization. AIAA Journal, 2019.

[32] James Reuther, Juan Alonso, Joaquim RRA Martins, and Stephen Smith. A coupled aero-structural optimization method for complete aircraft configurations. In 37th Aerospace Sciences Meeting and Exhibit, page 187, 1999.

[33] Matthieu Sacher, Régis Duvigneau, Olivier Le Maitre, Mathieu Durand, Elisa Berrini, Frédéric Hauville, and Jacques-André Astolfi. A classification approach to efficient global optimization in presence of non-computable domains. Structural and Multidisciplinary Optimization, 58:1537–1557, 2018.

[34] Mohammad H Sadraey. Aircraft design: A systems engineering approach. John Wiley & Sons, New Hampshire, USA, 2012.

[35] P. Saves, E. Nguyen Van, N. Bartoli, Y. Diouane, T. Lefebvre, C. David, S. Defoort, and J. Morlier. Bayesian optimization for mixed variables using an adaptive dimension reduction process: applications to aircraft design. AIAA SciTech, 2022.

[36] Robert Kyle Schmidt. The design of aircraft landing gear. SAE International, Warrendale, PA, 2021.

[37] Jeffrey P Slotnick, Abdollah Khodadoust, Juan Alonso, David Darmofal, William Gropp, Elizabeth Lurie, and Dimitri J Mavriplis. CFD vision 2030 study: a path to revolutionary computational aerosciences. No. NF1676L-18332, 2014.

[38] Ali Tfaily, Kenny Huynh, Pat Piperni, and Susan Liscouet-Hanke. Landing gear integration in an industrial multi-disciplinary optimization environment. SAE Technical Paper, 2013.

[39] Ali Tfaily and Michael Kokkolaras. Integrating air systems in aircraft multidisciplinary design optimization. In 2018 Multidisciplinary Analysis and Optimization Conference, page 3742, 2018.

[40] Egbert Torenbeek. Advanced aircraft design: conceptual design, analysis and optimization of subsonic civil airplanes. John Wiley & Sons, Hoboken, NJ, 2013.

[41] Anh Tran, Jing Sun, John M Furlan, Krishnan V Pagalthivarthi, Robert J Visintainer, and Yan Wang. pbo-2gp-3b: A batch parallel known/unknown constrained Bayesian optimization with feasibility classification and its applications in computational fluid dynamics. Computer Methods in Applied Mechanics and Engineering, 347:827–852, 2019.

[42] Christopher KI Williams and Carl Edward Rasmussen. Gaussian processes for machine learning. MIT press, Cambridge, MA, 2006.