Efficient Acquisition Functions for Bayesian Optimization in the Presence of Hidden Constraints

Ali Tfaily, Michael Kokkolaras, Nathalie Bartoli, Youssef Diouane

Efficient Acquisition Functions for Bayesian Optimization in the Presence of Hidden Constraints

Ali Tfaily*†
McGill University, Montreal, Quebec, Canada
Bombardier, Montreal, Quebec, Canada

Michael Kokkolaras‡
McGill University, Montreal, Quebec, Canada

Nathalie Bartoli§
ONERA/DTIS, Université de Toulouse, Toulouse, France

Youssef Diouane¶
Polytechnique Montréal, Montreal, Quebec, Canada

A common challenge in engineering design applications of Bayesian optimization is the presence of non-computable domains, which are often called hidden or unknown constraints. In these cases, the required function values cannot be computed in certain regions of the design space. In this work, we propose a novel method to handle hidden constraints by modifying a portion of the acquisition function of a typical Bayesian optimization algorithm using classifiers. Specifically, the proposed approach considers the exploitation part of the acquisition function, allowing the optimization algorithm to search unexplored regions of the design space. Convergence properties are maintained regardless of the type of classifier used. Results obtained for numerical experiments on analytical test problems confirm the usefulness of the proposed method of handling hidden constraints in the context of Bayesian optimization.

I. Nomenclature

BO	=	Bayesian optimization
C_{nf}	=	predicted class of the simulation between failure and non-failure
EI	=	expected improvement acquisition function
EFI	=	expected feasible improvement acquisition function
EFI_{FE}	=	feasibility enhanced expected feasible improvement acquisition function
`GPC`	=	Gaussian process classifier
`kNN3`	=	k-nearest neighbors classifier with $k = 3$
`SVM`	=	support vector machine
Nan	=	Not a number Python object
PI	=	probability of improvement acquisition function
p_{nf}	=	probability of non-failure of the simulation
SBO	=	surrogate-based optimization
WB2S	=	scaled Watson-Barnes acquisition function
$y_{\min}$	=	minimum observed value of objective function
$\hat{y}(x)$	=	mean value of objective function surrogate model
$\hat{s}(x)$	=	standard deviation value of objective function surrogate model
Φ	=	normal cumulative distribution function

*PhD Candidate, Department of Mechanical Engineering, AIAA Student Member
†Aircraft Systems Chief Architect, Advanced Product Development Department
‡Professor, Department of Mechanical Engineering, AIAA Associate Fellow and MDO TC Member
§Senior Researcher, Department of Information Processing and Systems, AIAA MDO TC Member
¶Assistant Professor, Department of Mathematical and Industrial Engineering, AIAA Member

ϕ = normal probability density function
α = acquisition function exploration factor for failure regions of the design space.

II. Introduction

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 adapted during the optimization process. In this work, we consider the Bayesian optimization (BO) paradigm where Gaussian processes are used to model the objective and constraint functions. BO has become a popular method for solving optimization problems in aerospace engineering design [1–4].

Optimization in engineering design problems often involves failed ("crashed") simulations that prevent the completion of an optimization. A failed simulation during the optimization process is defined as a simulation that terminates unexpectedly resulting in unsuccessful computation of objective or constraint function values. Such failed simulations are often referred to as hidden or unknown constraints. Hidden or unknown constraints are not known explicitly to the optimization algorithm [5].

Handling hidden constraints in SBO algorithms has been investigated in the literature. Lee *et al.* used a random forest classifier to calculate a feasible probability and integrated the classifier within an expected improvement (EI) acquisition function [6]. In [7], Sacher *et al.* extended a method developed by Basudhar *et al.* in [8]: they use a least-squares support vector machine technique for classification of both known and hidden constraints which is used to model the boundary of the feasible design space. Gelbart *et al.* proposed a framework to perform Bayesian optimization under hidden constraints using a probabilistic approach where constraint satisfaction can be determined by meeting a probability of feasibility threshold [9] . Antonio presented a sequential SBO framework for problems that are either undefined or partially outside the feasible region [10]. His 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. Tran *et al.* proposed the use of an external classifier using Gaussian processes that determines a probability of feasibility [11]. The calculated probability is then used to condition the acquisition function directly, similar to the approach proposed in [6]. Bachoc *et al.* developed a Gaussian process classifier and applied it to a modified EI acquisition function [12]. Audet *et al.* proposed the use of k-nearest neighbor classifiers to build surrogate models that guide the mesh adaptive direct search optimization algorithm [13].

A drawback of these methods is that classifier inaccuracies in the early stages of the optimization process may impact the latter adversely. Therefore, we propose a new formulation of acquisition functions and consider different classifiers to handle hidden constraints efficiently. The proposed approach entails modifying acquisition functions to enable exploration of the design space where the classifier used to predict failure may be inaccurate.

The outline of the paper is as follows. The theoretical background of Bayesian optimization and acquisition functions is reviewed briefly in Section III. The proposed acquisition function is defined in Section III.B. Modeling of hidden constraint classifiers is presented in Section III.C. Numerical experiments are conducted in Section IV. Conclusions and perspectives of this work are drawn in Section V.

III. Bayesian Optimization

Consider the constrained optimization problem

$$\begin{aligned} \min_{x \in \Omega} \quad & y(x) \\ \text{subject to} \quad & c_i(x) \leq 0, \; i = 1, \ldots, m, \end{aligned} \tag{1}$$

where the objective function y and the constraint functions c_i are generally evaluated by means of blackboxes, i.e., computational models whose structure are not accessible by the design engineer.

In surrogate-based optimization, the objective and constraint functions are evaluated using surrogate models such as Gaussian processes. In Bayesian optimization, these are adapted during the optimization process. Prior to the optimization, a design of experiments (DOE) is typically conducted to generate sites that sample the design space. The higher-fidelity models are evaluated at these sites to generate the data used to build the surrogate models and their probability distributions, known as priors. The optimization loop is described in Algorithm 1; it entails the adaptation of the surrogate models and the solution of a sequential enrichment problem, which aims at maximizing an acquisition

function that balances exploration and exploitation. The high fidelity models are used to evaluate the objective and constraint functions at the solution of the sequential enrichment problem and the new data are used to update the surrogates. This process continues until a certain convergence criterion is met or until a maximum number of iterations is reached.

Algorithm 1 Constrained Bayesian optimization

input initial surrogate models;

1: **for** $i = 1$ **to** max_{it} **do**
2: Update surrogate models of objective and constraint functions using Gaussian processes;
3: Solve a sequential enrichment problem by maximizing a selected acquisition function to find x^{i+1}
4: Use high-fidelity models to evaluate objective and constraint functions at x^{i+1}
5: **If** the convergence criterion is met **then** Stop
6: **end for**

output The best feasible point

A. Acquisition Functions

Several acquisition functions have been proposed and investigated, ranging from traditional and widely used functions such as probability of improvement (PI) [14] and expected improvement (EI) [15] to newly developed functions such as scaled Watson Barnes (WB2S) [16]. The probability of improvement [14] is given by

$$\mathrm{PI}(x) = \mathbb{P}(y(x) \leq y_{\min}) = \Phi\left(\frac{y_{\min} - \hat{y}(x)}{\hat{s}(x)}\right) \tag{2}$$

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) [15] is of the form:

$$\mathrm{EI}(x) = \begin{cases} (y_{\min} - \hat{y}(x))\,\Phi\left(\dfrac{y_{\min} - \hat{y}(x)}{\hat{s}(x)}\right) + \hat{s}(x)\,\phi\left(\dfrac{y_{\min} - \hat{y}(x)}{\hat{s}(x)}\right), \\ 0, \quad \text{if } \hat{s} = 0, \end{cases} \tag{3}$$

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.

B. Enhanced Acquisition Function for Hidden Constraints

Previous methods adapted BO to handle hidden constraints through conditioning the acquisition function by either the probability of non-failure or class of non-failure [6, 7, 9, 11, 12]. Another approach aims at constraining the design space of an optimization based on regions of predicted failures [7, 8, 10]. The expected feasible improvement acquisition functions associated with the aforementioned approaches are formulated as

$$\mathrm{EFI}_{\mathbf{P}}(x) = p_{\mathrm{nf}}(x)\ \mathrm{EI}(x) \tag{4}$$

and

$$\mathrm{EFI}_{\mathbf{C}}(x) = c_{\mathrm{nf}}(x)\ \mathrm{EI}(x), \tag{5}$$

where p_{nf} is the probability of non-failure and c_{nf} is the class of non-failure calculated using some surrogate model Z. The main drawback of these 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 away from exploration regions of the design space, especially if a non-sequential framework is used. In addition, global convergence cannot always be guaranteed using the expected feasible improvement acquisition functions in Eq. (4) and Eq. (5) for all types of classifiers. Global convergence for Eq. (4) has been shown using a Gaussian process classifier in [12]; however, this cannot be always guaranteed for other types of classifiers.

We therefore propose a new acquisition function that conditions its exploitation term and reduces the impact on its exploration term by means of an exploration factor α. We named it feasibility-enhanced expected improvement

acquisition function ($\mathrm{EFI_{FE}}$); it aims at handling hidden constraints both in terms of exploitation (obtaining a better value of y) and exploration of the feasible domain. It is formulated as

$$\mathrm{EFI_{FE}}(x) = \begin{cases} p_{\mathrm{nf}}(x)\,(y_{\min} - \hat{y}(x))\,\Phi\left(\dfrac{y_{\min} - \hat{y}(x)}{\hat{s}(x)}\right) + p_{\mathrm{nf}}(x)^{\alpha}\hat{s}(x)\phi\left(\dfrac{y_{\min} - \hat{y}(x)}{\hat{s}(x)}\right), \\ 0, \quad \text{if } \hat{s} = 0, \end{cases} \tag{6}$$

where $\alpha \in [0, 1]$ is the previosly mentioned exploration factor that allows the acquisition function to approach failure regions of the design space. The $\mathrm{EFI_{FE}}(x)$ acquisition function conditions the exploitation portion of $\mathrm{EI}(x)$ similar to $\mathrm{EFI_P}(x)$; however the impact on the exploration portion is minimized by the term α except when the probability of non-failure $p_{\mathrm{nf}}(x)$ is close to zero.

C. Modeling of Hidden Constraints using Supervised Machine Learning Classifiers

Several models have been proposed in the literature to represent hidden constraints using supervised machine learning techniques. The use of Gaussian process classifiers, conditioned by signs of observations was proposed in [12]. Random forests were used in [6]. We consider additional classification means, including nearest neighbor classifiers, decision trees and rule-based classifiers, probabilistic models, and support vector machines. We use labeled data To construct and adapt these classifiers.

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 [17]. 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

$$p_{\mathrm{nf}}(x) = \frac{\sum_{i \in N} I(x) d(x, x_i)}{\sum_{i \in N} d(x, x_i)}, \tag{7}$$

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 [18].

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 [19]. 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

$$p_{\mathrm{nf}}(x) = \frac{1}{N_m} \sum_{x \in N_m} I(x). \tag{8}$$

Probabilistic classifiers such as logistic regression construct a relationship between the input features and output class as a probability [17]. 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 [17, 19]

$$p_{\mathrm{nf}}(x) = \frac{1}{1 + e^{(\theta_0 + \theta^{\top} x)}}, \tag{9}$$

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. (9).

Non-linear support vector machines (SVM) classify instances by defining a boundary that separates classes of samples from a dataset. In a binary classification problem, a SVM model predicts the class $sign(F)$ of an instance x from [17] using data from N training samples

$$F(x) = b + \sum_{i=1}^{N} \lambda_i y_i K(x_i, x), \tag{10}$$

where b is a bias, 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 , y_i is the class of

the training sample, and K is a kernel function used to handle non-linearity in the defined boundary. Several kernel functions can be used; a popular choice is the Gaussian radial basis function kernel [10]

$$K(x_i, x) = (x_i^\top x + 1)^p, \tag{11}$$

where p is the kernel polynomial degree. The polynomial kernel used in [8] is defined by

$$K(x_i, x) = e^{-\frac{||x_i - x||^2}{2\sigma^2}}, \tag{12}$$

where σ^2 is the hyper-parameter of the Gaussian radial basis function kernel that determines the similarity of x and x_i.

In this work, we use the Gaussian radial basis function kernel for SVM modeling. SVM does not directly compute probability to obtain class predictions. We present the method used in [18] to compute this probability. In a case of two class classification, the class probabilities are calibrated using the scaling proposed in [20], where logistic regression is performed on the SVM's scores similar to Eq. (9).

Gaussian processes can be used as classifiers using Laplace approximation over $\mathbb{R}$ [21]:

$$p_{\text{nf}}(x) = \int \sigma(\hat{y}(x)) q(\hat{y}(x)) d\hat{y}(x), \tag{13}$$

where σ is a sigmoid function and q is a Gaussian with a mean of $\hat{y}$.

IV. Numerical Experiments

The proposed acquisition function with the considered classifiers were coded in an open-source Python Bayesian optimization tool [22] Scikit-learn libraries [18]. We conducted numerical experiments using four examples with different problem sizes.

A. Examples

The first numerical test case is an unconstrained optimization problem inspired by the test case presented in [6]:

$$\begin{aligned} &\min_{x \in \mathbb{R}^2} y(x) \\ &\text{with } \; y(x) = -w(x_1) w(x_2), \\ &\text{where } \; w(z) = e^{-(z-1)^2} + e^{-0.8(z+1)^2} - 0.5 \sin(8(z + 0.1)) \end{aligned} \tag{14}$$

We confined $x_1, x_2 \in [-2, 2]^2$. A hidden failure region inside of an ellipse returns non-valid (i.e., NaN) to represent a hidden constraint as shown in the top middle graph of Fig. 1, where the dark-shaded (blue) region represents the region where simulation failure occurs. The masked function then overlays the hidden constraint over the objective function, hiding thus a portion of the design space. Without the hidden constraint, the function has a global minimum at $x_1 = x_2 = -1.04$. However, with the introduction of the failure region (hidden constraint), 2 local minima are feasible on each side of the ellipse with the new minimum value being -1.09 at (-1.04, 1.14) and (1.14, -1.04). As a first example a k-nearest neighbor classifier is used with $k = 3$. The acquisition function exploration factor α was set to 0.3. It is noted that the behaviour of the hidden constraint estimate is captured in the EFI_{FE} acquisition function and the new minima regions are identified with higher values of the acquisition function.

Firstly, the newly developed acquisition function behaviour is validated by comparing it to the EI and EFI_{P} acquisition functions using a DOE of 50 points. Figure 2 shows that the new method improves exploration of the acquisition function especially in regions close to the hidden constraints. It is obvious that the use of an unconditioned EI acquisition function without the use of a classifier (left graph in Fig. 2) leads to an incorrect model of the design space where the higher values of the acquisition function (red contours) are within the failure region of the design space. The middle and right graphs of Fig. 2 then compare EFI_{P} in Eq. (4) to the proposed method EFI_{FE} in Eq. (6) using a kNN classifier with k = 3. It is noted the proposed method favors exploration in the top left region of the graph where another minimum is found.

The second numerical test case is based on the Branin function [23] with a failure region defined by

$$\begin{aligned} &\min_{x \in \mathbb{R}^2} \; y(x) \\ &y(x) = \left(\bar{x}_2 - \frac{5.1}{4\pi^2} \bar{x}_1^2 + \frac{5}{\pi \bar{x}_1} - 6 \right)^2 + 10 \left(1 - \frac{1}{8\pi} \right) \cos \bar{x}_1 + 10, \end{aligned} \tag{15}$$

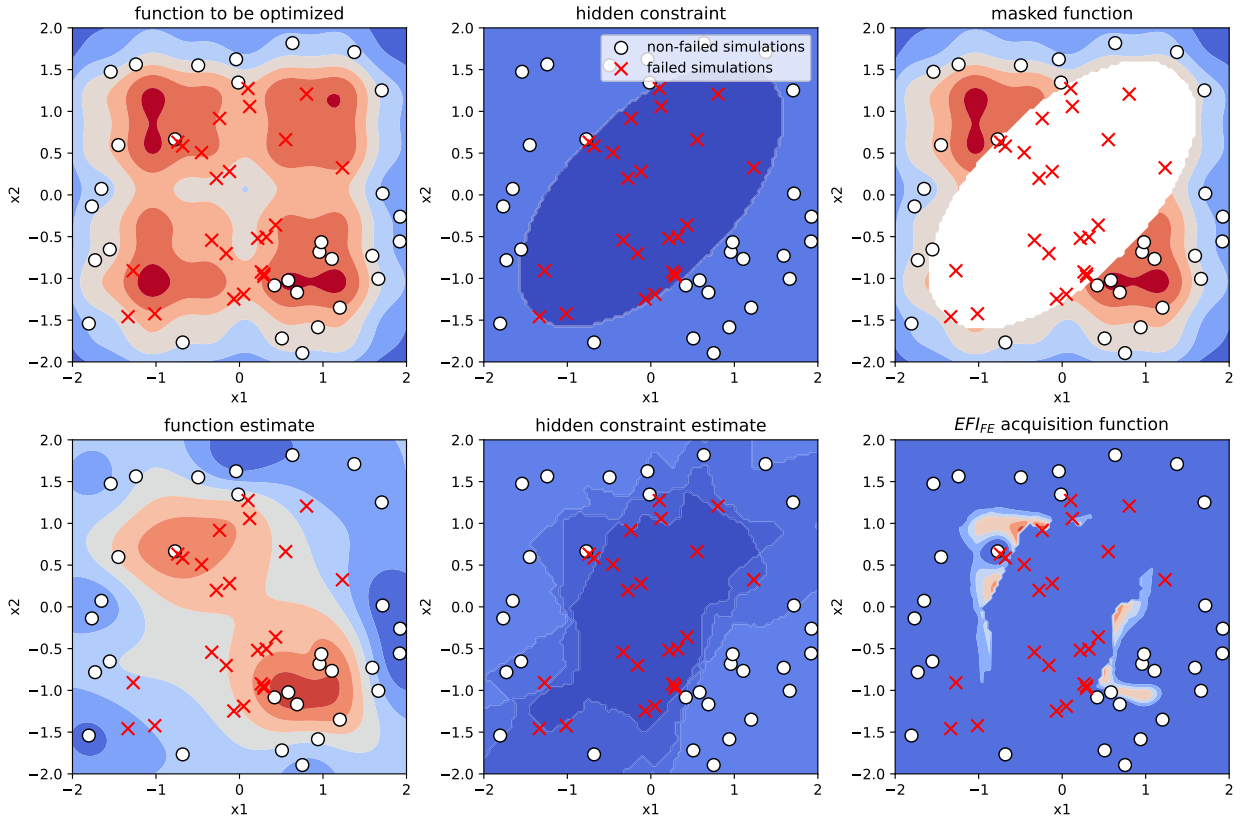


Fig. 1 Top left: negative objective function; top middle: hidden constraint; top right: negative function masked with hidden constraint; bottom left: GP estimate; bottom middle: hidden constraint classifier estimate; and bottom right: acquisition function for the problem in Equation (14) using a DOE of 50 points. Red contours represent higher values and blue contours represent lower values.

where $\bar{x}_1 = 15x_1 - 5$, $\bar{x}_2 = 15x_2$, and $x_1, x_2 \in [0, 1]$. A hidden region returns a code failure acting as a hidden constraint when $|x_1 - 0.5| < 0.5$ and $|x_2 - 0.5| < 0.4$.

The third numerical test case is the 4-dimensional Rosenbrock function minimization problem defined by

$$\begin{aligned} &\underset{x \in \mathbb{R}^4}{\text{minimize}}\ y(x) \\ &y(x) = \sum_{i=1}^{3} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2], \end{aligned} \tag{16}$$

where $x_i \in [-2.048, 2.048]$, for all $i = 1, ..., 4$. A failure region is identified when $0 < x_i < 1$ for $i = 1, 2$ and $1 < x_i < 2$ for $i = 3, 4$.

The fourth and last problem is 6-dimensional; it is used to test the approach based on the Schwefel function [24] defined by

$$\begin{aligned} &\min_{x \in \mathbb{R}^6}\ y(x) \\ &y(x) = 2513.895 - \sum_{i=1}^{6} x_i \sin\sqrt{|x_i|}, \end{aligned} \tag{17}$$

where $x_i \in [-500, 500]$, for all $i = 1, ..., 6$. A failure region is identified when $350 < x_i < 420$ for $i = 1..., 6$. The failure region is close to the global minimum located at $x_i = [420.9687]$ for all $i = 1, ..., 6$ to test the capability of EFI_{FE} in problems where the hidden constraint is close to the global minimum.

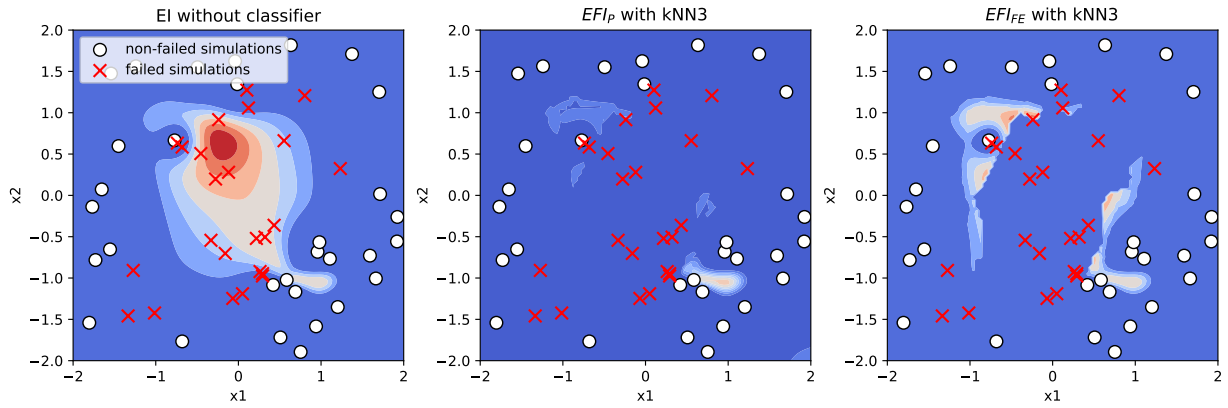


Fig. 2 Comparison of acquisition functions for the problem in Equation (14) using a DOE of 50 points. Red contours represent higher values and blue contours represent lower values.

B. Sensitivity Analysis of Key Parameters and Classifiers

The impact of the exploration factor α on optimization convergence using the four numerical test case from Eqs. (14), (15), (16), and (17) is assessed. Each test case is run 20 times using a common DOE of 10 points for different values of α and 90 optimization iterations. Figure 3 depicts the convergence rate using four different values of α: 0.3, 0.1, 0.02, and 0.01. The selected values are all below 0.5 to highlight the impact of the exploration factor since a higher value of α leads to closer results to EFI_P where at $\alpha = 1$, $\text{EFI}_\text{FE} = \text{EFI}_\text{P}$. Comparison of results between EFI_FE and EFI_P is shown in Section IV.C. It is noted from Fig. 3 that for all test cases, the minimum was practically achieved for all considered values of α, demonstrating that the impact of α is not significant. We use $\alpha = 0.3$ for the remainder of the numerical investigations.

Different types of classifiers were also tested by comparing convergence results of the four numerical test cases similarly by running 20 different optimizations with the same initial DOE of 10 points and a total budget of 100 objective function evaluations. The six tested classifiers, as detailed in Section III.C, are:

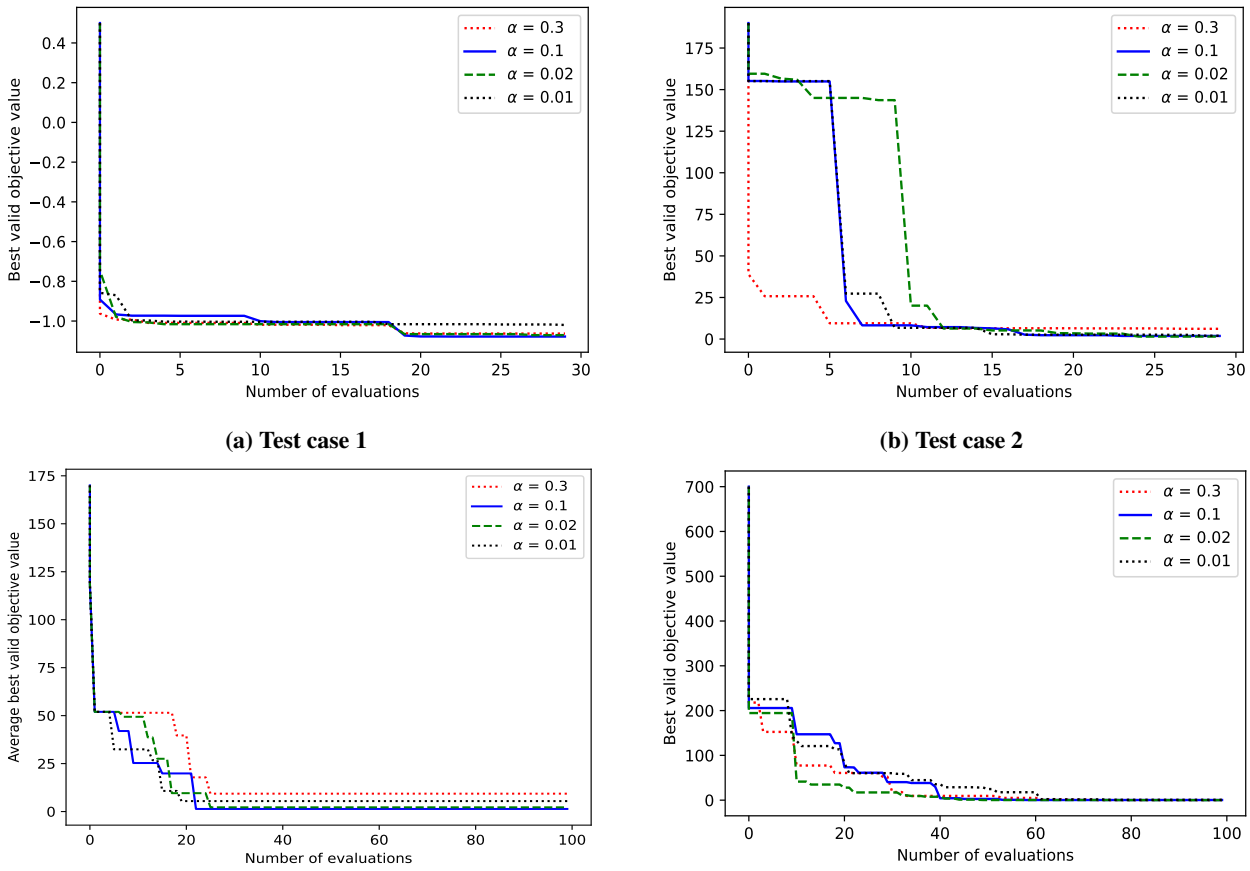
1) a k-nearest neighbors classifier with k=3 (`kNN3`),
2) an SVM classifier (`SVM`),
3) a Gaussian process classifier (`GPC`),
4) a decision tree classifier (`DecisionTree`),
5) a random forest classifier (`RandomForest`), and
6) a logistic regression classifier (`LogisticRegression`).

Figure 4 shows that `kNN3` and `SVM` classifiers produce the best convergence rates consistently for the four test cases. For that reason, in the remainder of this paper, a `kNN3` classifier will be used.

C. Comparison of Results

Optimization results of the new method EFI_FE versus EFI_P are compared for the four numerical test cases. Each test case is compared using 20 optimization runs with common DOE of 10 points and 90 iterations of optimization. Results are compared by showing convergence rates and box plots of the best valid objective values for the 20 optimization runs for both EFI_FE and EFI_P.

For test case 1, Fig. 5 shows that EFI_FE produces better convergence results than EFI_P for all of the optimization runs. For test case 2, comparing the convergence of optimization results between EFI_FE with EFI_P in Fig. 6 (a), it is noted that both acquisition functions reach the minimum. However, comparing the scatter of optimization minima for each of the acquisition functions shows that EFI_FE results are more robust as shown in Fig. 6 (b). Similarly, results of test case 3 (see Fig. 7) show that the convergence and minimum scatter are improved when using the EFI_FE acquisition function over EFI_P. Similar to previous test cases, the masked Schwefel function in test case 4 (see Fig. 8) showed that EFI_FE improved the convergence speed and lowered the achieved minimum compared to EFI_P.



(a) Test case 1

(b) Test case 2

(c) Test case 3

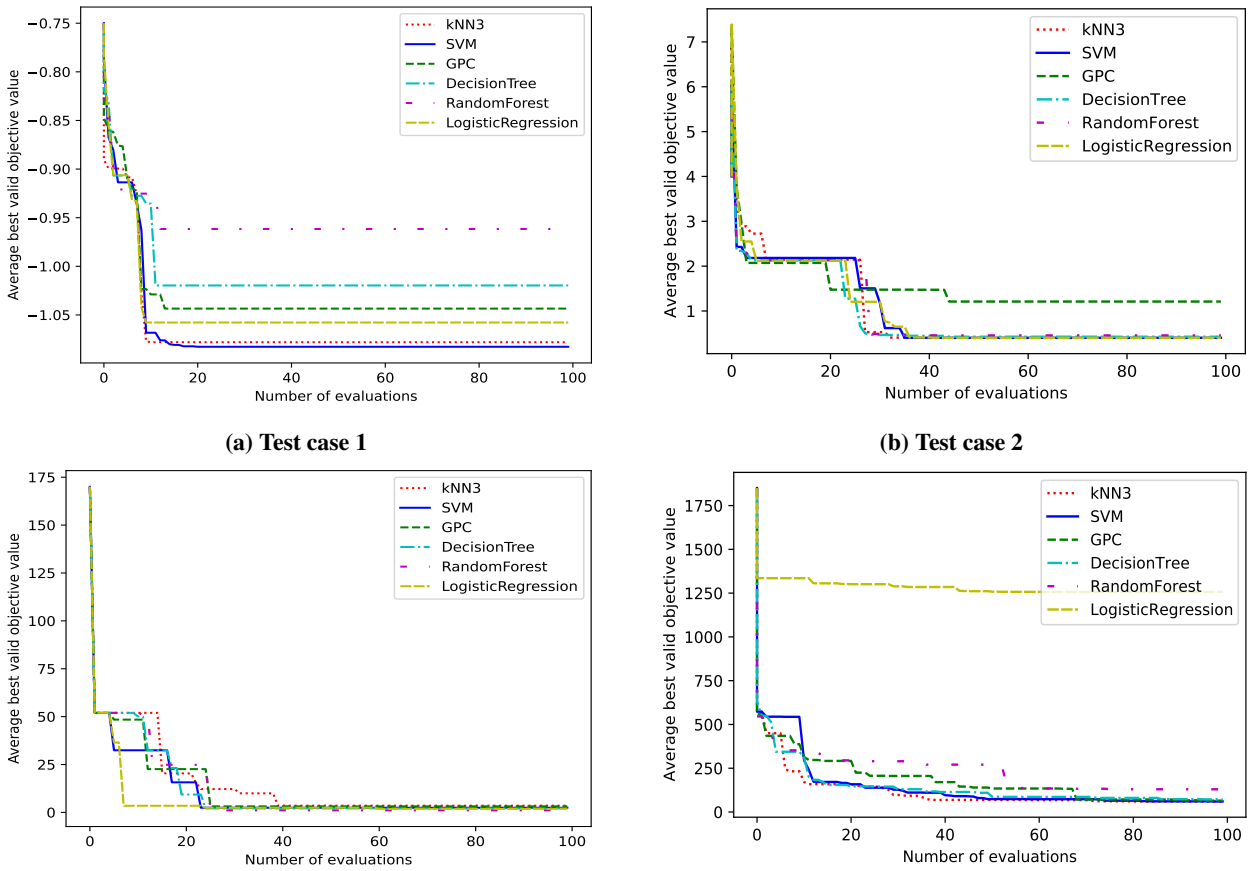
(d) Test case 4

Fig. 3 A sensitivity analysis (average of 20 runs) with respect to the choice of α.

V. Conclusion and Perspectives

We proposed a new acquisition function to handle hidden constraints in Bayesian optimization. Our approach aims at enabling exploration of the design where the classifier used to predict failure may be inaccurate. The proposed method was tested on four analytical test cases with problem size ranging from 2 to 6, showing benefits over existing methods and thus great potential.

Including constraints and mixed-categorical design variables will allow to solve more realistic conceptual aircraft design problems, e.g., [4]. In this context, inspired by the recent software developments [25, 26], an extension of the method to solve realistic aircraft design optimization problems shall be investigated in a near future.



(a) Test case 1 **(b) Test case 2**

(c) Test case 3 **(d) Test case 4**

Fig. 4 A sensitivity analysis (average of 20 runs) with respect to the choice of classifier.

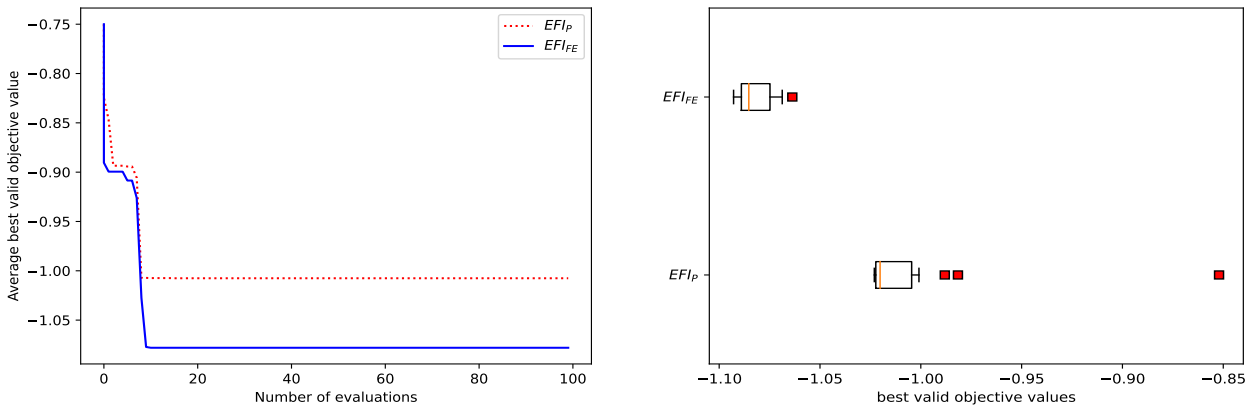
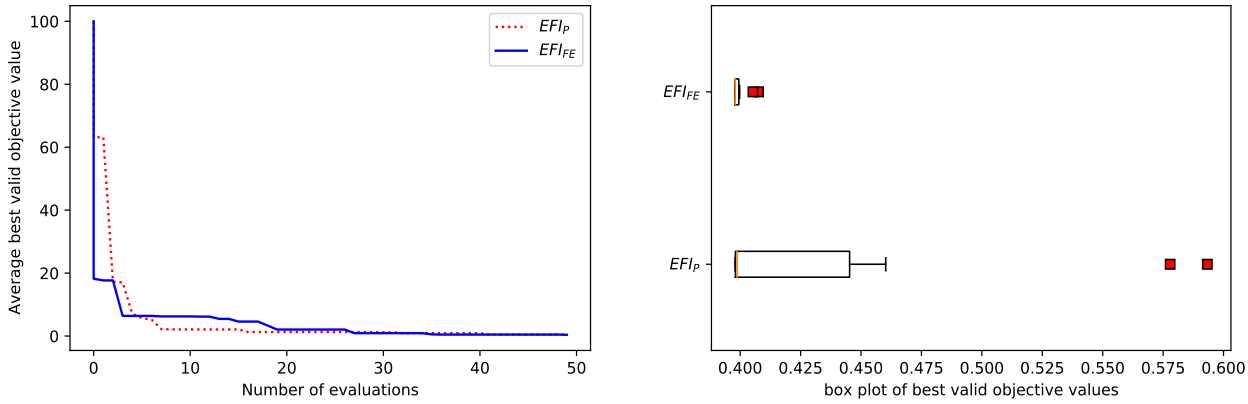
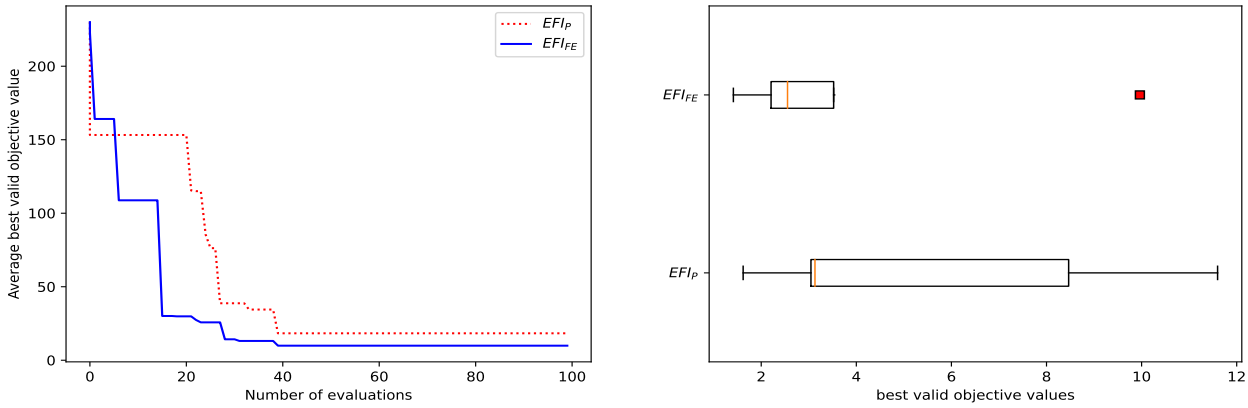


Fig. 5 Optimization results (using 20 runs) for the first test case using failure classifier kNN with $k = 3$.



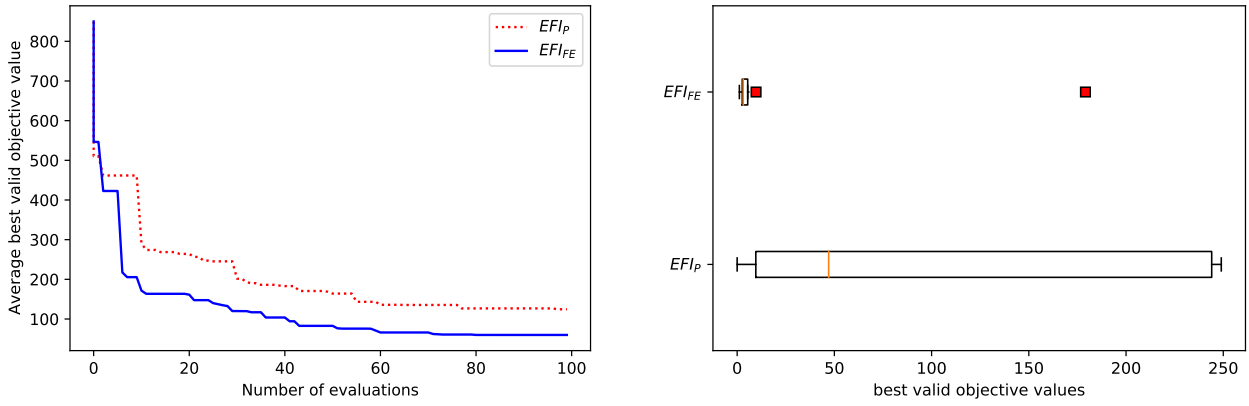
(a) Convergence plots **(b) Box plot at the optimum.**

Fig. 6 Optimization results (using 20 runs) on the second test case using failure classifier kNN with $k = 3$.



(a) Convergence plots **(b) Box plot at the optimum.**

Fig. 7 Optimization results (using 20 runs) on the third test case using failure classifier kNN with $k = 3$.



(a) Convergence plots **(b) Box plot at the optimum.**

Fig. 8 Optimization results (using 20 runs) on the fourth test case using failure classifier kNN with $k = 3$.

References

[1] Lam, R., Poloczek, M., Frazier, P., and Willcox, K. E., "Advances in Bayesian optimization with applications in aerospace engineering," *2018 AIAA Non-Deterministic Approaches Conference*, 2018, p. 1656.

[2] Priem, R., Gagnon, H., Chittick, I., Dufresne, S., Diouane, Y., and Bartoli, N., "An efficient application of Bayesian optimization to an industrial MDO framework for aircraft design." *AIAA AVIATION 2020 FORUM*, 2020, p. 3152.

[3] Jim, T. M., Faza, G. A., Palar, P. S., and Shimoyama, K., "Bayesian optimization of a low-boom supersonic wing planform," *AIAA journal*, Vol. 59, No. 11, 2021, pp. 4514–4529.

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

[5] Digabel, S. L., and Wild, S. M., "A taxonomy of constraints in simulation-based optimization," *arXiv preprint arXiv:1505.07881*, 2015.

[6] Lee, H., Gramacy, R., Linkletter, C., and Gray, G., "Optimization subject to hidden constraints via statistical emulation," *Pacific Journal of Optimization*, Vol. 7, No. 3, 2011, pp. 467–478.

[7] Sacher, M., Duvigneau, R., Le Maitre, O., Durand, M., Berrini, E., Hauville, F., and Astolfi, J.-A., "A classification approach to efficient global optimization in presence of non-computable domains," *Structural and Multidisciplinary Optimization*, Vol. 58, No. 4, 2018, pp. 1537–1557.

[8] Basudhar, A., Dribusch, C., Lacaze, S., and Missoum, S., "Constrained efficient global optimization with support vector machines," *Structural and Multidisciplinary Optimization*, Vol. 46, No. 2, 2012, pp. 201–221.

[9] Gelbart, M. A., Snoek, J., and Adams, R. P., "Bayesian optimization with unknown constraints," *arXiv preprint arXiv:1403.5607*, 2014.

[10] Antonio, C., "Sequential model based optimization of partially defined functions under unknown constraints," *Journal of Global Optimization*, 2019, pp. 1–23.

[11] Tran, A., Sun, J., Furlan, J. M., Pagalthivarthi, K. V., Visintainer, R. J., and Wang, Y., "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*, Vol. 347, 2019, pp. 827–852.

[12] Bachoc, F., Helbert, C., and Picheny, V., "Gaussian process optimization with failures: classification and convergence proof," *Journal of Global Optimization*, Vol. 78, No. 3, 2020, pp. 483–506.

[13] Audet, C., Caporossi, G., and Jacquet, S., "Binary, unrelaxable and hidden constraints in blackbox optimization," *Operations Research Letters*, Vol. 48, No. 4, 2020, pp. 467–471.

[14] Kushner, H. J., "A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise," 1964.

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

[16] Bartoli, N., Lefebvre, T., Dubreuil, S., Olivanti, R., Priem, R., Bons, N., Martins, J. R., and Morlier, J., "Adaptive modeling strategy for constrained global optimization with application to aerodynamic wing design," *Aerospace Science and technology*, Vol. 90, 2019, pp. 85–102.

[17] Aggarwal, C. C., *Data mining: the textbook*, Springer, 2015.

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

[19] Hastie, T., Tibshirani, R., Friedman, J. H., and Friedman, J. H., *The elements of statistical learning: data mining, inference, and prediction*, Vol. 2, Springer, 2009.

[20] Platt, J., et al., "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," *Advances in large margin classifiers*, Vol. 10, No. 3, 1999, pp. 61–74.

[21] Williams, C. K., and Rasmussen, C. E., *Gaussian processes for machine learning*, Vol. 2, MIT press Cambridge, MA, 2006.

[22] Nogueira, F., "Bayesian Optimization: Open source constrained global optimization tool for Python," , 2014–. URL `https://github.com/fmfn/BayesianOptimization`.

[23] Picheny, V., Wagner, T., and Ginsbourger, D., "A benchmark of kriging-based infill criteria for noisy optimization," *Structural and multidisciplinary optimization*, Vol. 48, 2013, pp. 607–626.

[24] Laguna, M., and Marti, R., "Experimental testing of advanced scatter search designs for global optimization of multimodal functions," *Journal of Global Optimization*, Vol. 33, 2005, pp. 235–255.

[25] Saves, P., Diouane, Y., Bartoli, N., Lefebvre, T., and Morlier, J., "A mixed-categorical correlation kernel for Gaussian process," *arXiv preprint arXiv:2211.08262*, 2022.

[26] Rufato, R. C., Diouane, Y., Henry, J., Ahlfeld, R., and Morlier, J., "A mixed-categorical data-driven approach for prediction and optimization of hybrid discontinuous composites performance," *AIAA AVIATION 2022 Forum*, 2022.