Journal of Global Optimization (2021) 79:281–303
https://doi.org/10.1007/s10898-019-00860-4

Sequential model based optimization of partially defined functions under unknown constraints

Candelieri Antonio[1]

Received: 21 January 2019 / Accepted: 18 November 2019 / Published online: 2 December 2019

Abstract
This paper presents a sequential model based optimization framework for optimizing a black-box, multi-extremal and expensive objective function, which is also partially defined, that is it is undefined outside the feasible region. Furthermore, the constraints defining the feasible region within the search space are unknown. The approach proposed in this paper, namely SVM-CBO, is organized in two consecutive phases, the first uses a Support Vector Machine classifier to approximate the boundary of the unknown feasible region, the second uses Bayesian Optimization to find a globally optimal solution within the feasible region. In the first phase the next point to evaluate is chosen by dealing with the trade-off between improving the current estimate of the feasible region and discovering possible disconnected feasible sub-regions. In the second phase, the next point to evaluate is selected as the minimizer of the Lower Confidence Bound acquisition function but constrained to the current estimate of the feasible region. The main of the paper is a comparison with a Bayesian Optimization process which uses a fixed penalty value for infeasible function evaluations, under a limited budget (i.e., maximum number of function evaluations). Results are related to five 2D test functions from literature and 80 test functions, with increasing dimensionality and complexity, generated through the Emmental-type GKLS software. SVM-CBO proved to be significantly more effective as well as computationally efficient.

Keywords Sequential model based optimization · Constrained global optimization · Partially defined objective functions

1 Introduction

Sequential model based optimization (SMBO), and more precisely Bayesian Optimization (BO) [1], is a global optimization approach which has been shown to be effective and sample efficient in the case of black-box expensive objective functions [2–11]. Indeed, it is currently the standard approach, in the machine learning community, for solving automatic algorithm configuration [12], hyper-parameter tuning [13–16] and Neural Architecture Search [13, 17]. Its building blocks are a probabilistic surrogate model—usually

✉ Candelieri Antonio
candelieriantonio@gmail.com; antonio.candelieri@unimib.it

[1] University of Milano-Bicocca, Milan, Italy

a Gaussian Process (GP)—used to approximate the objective function and an acquisition function to provide the next promising point to evaluate, balancing between exploitation and exploration.

Although BO has been used to solve global optimization problems in bounded-box search spaces [1–3, 7, 8, 18], more interesting and realistic cases are related to constrained global optimization (CGO) [19–21], where constraints are not defined as simple limits on the values of the variables spanning the search space. The traditional way of solving the constrained problems consists in solving an unconstrained problem where the original objective function is penalized depending on the violation of feasibility. For some classes of functions, a finite constant penalty can provide the same solution of using a variable penalty function [22, 23]. More recently, a derivative-free extension of [23], based on DIRECT, has been proposed in [24] and [25] making the approach suitable for globally solving constrained problems where the derivatives are not available. Another derivative-free approach, based on Differential Search, has been proposed in [26], based on the idea of exact penalty function and using a dynamical penalty factor to achieve a better trade-off between exploration and exploitation. The main drawback of penalty-based approaches is that a suitable choice of the penalty constant is rather difficult. In the case the penalty constant is not large enough the global optimizer can fall out of feasible domain, while if it is too large it, the penalized function has worse properties than the initial one, for instance Lipschitz constant can increase significantly. However, when some a priori information about the slope of the function is known, deterministic approaches have been also applied to solve CGO problems, such as in [22], where Lipschitz condition for the objective function and computable boundaries for the constraints are assumed. Optimization is then performed by reducing the initial multidimensional problem to a family of one-dimensional sub-problems, each one solved through univariate Lipschitz optimization.

A specific case in CGO is related to *partially defined objective functions*, meaning that the objective function is undefined outside the feasible region [19, 27–29] and cannot be therefore computed. Thus, for each point x belonging to the search domain, a pair $(y, f(x))$ is observed. For sake of homogeneity with the rest of the paper, let denote by $y = 1$ the case that $f(x)$ is defined/computable (i.e., x feasible), $y = -1$ otherwise. Other notations are possible (for instance the one used in [30]) but, in any case, the following assumption is considered: it is as costly to check if an evaluation at x fails as to observe the value of $f(x)$ when there is no failure. Partially defined functions are known with different names in different scientific communities; for instance, in [30] the terms *simulation failures* and *crash constraints* are used. In the quoted reference, the optimization of a computer model is considered, where each run either fails or returns a valid output (i.e., objective function value). A joint GP model for classification of the inputs (failure or success) and for regression of the objective function is proposed. Another definition is *non-computable domains*, as introduced in [31], referring to situations occurring when the search space encompasses points corresponding to an unphysical configuration, an ill-posed problem, or a non-computable problem due to the limitation of numerical solvers. The set of evaluated points is split into two subsets corresponding to computable and non-computable points, respectively (aka, feasible/infeasible). A surrogate model for the objective function is built using the set of computable points, only, whereas a probabilistic classification model is built using all the points, where each point belongs to one of the two classes, computable and non-computable, respectively. The aim of the classifier is to avoid proposing new points in the non-computable domain during the sequential optimization procedure.

Finally, a further challenging issue, for CGO problems, arises when constraints are unknown and the—consequently unknown—feasible portion of the search space cannot be analytically expressed. This property of the problem is independent by the nature of the objective function: it can or cannot be partially defined.

A useful taxonomy of constraints has been proposed in the simulation–optimization community [32], highlighting the significant difference between the types of constraints encountered in black-box and simulation-based optimization from those treated in nonlinear programming. The taxonomy is denoted QRAK, where the acronym corresponds to four properties of the constraints:

- Q is for Quantifiable. A quantifiable constraint is a constraint for which the violation can be quantified (otherwise it is named Nonquantifiable).
- R is for Relaxable. A relaxable constraint is a constraint that does not need to be satisfied to compute the objective (otherwise it is Unrelaxable).
- A is for A priori. An a priori constraint is a constraint for which feasibility can be confirmed without running a simulation (otherwise it is named Simulation constraint).
- K is for Known. A known constraint is a constraint that is explicitly given in the problem formulation (otherwise it is named Hidden). A hidden constraint typically (but not necessarily) appears when the simulation crashes. For such constraints, one can detect only violations, typically when some error flag or exception is raised. This concept is clearly another term for *partially defined objective functions*, *simulation failures*, *crash constraints* or *non-computable domains*.

According to this taxonomy, Simulation constraints (i.e., not A priori) correspond to the definition of *unknown constraints* adopted by many relevant prior works in the BO community [33–37]. Some of these references propose new acquisition functions, such as Integrated Expected Conditioned Improvement (IECI) [38] and a modification of the Predictive Entropy Search (PES) [39]. Most of them uses independent GPs to model the objective function and the constraints, requiring two strong assumptions: a priori knowledge about the number of constraints and the independence among objective function and all the constraints. The assumption of independence permits to compute a "probability of feasibility" simply as the product of individual probability of feasibility for each constraint. The result is multiplied to the acquisition function, usually Expected Improvement (EI), in order to optimize the objective function while satisfying, with high probability, the constraints. Basically, it is a CGO with a penalty function represented by the probability of feasibility.

The main contribution of this paper is the development of a method which does not require any of the two previous assumptions.

Finally, although it is not addressed in this paper, it is important to also mention Safe BO [40, 41], a closely related topic to CGO, where some the objective function is constrained to do not violate a "safety threshold" [40] or a "safety area" [41]. Thus, any new function evaluation must be performed at "safe points" only, where the difficulty is to ensure that the safety will be satisfied at any point during the optimization process. In this case, safe region and feasible region can be considered synonym: the important difference with respect to CGO is that Safe BO does not allow—at least with a given probability—any function evaluations outside the feasible/safe region.

The idea proposed in this this paper is to use Support Vector Machine (SVM) [42, 43] to sequentially estimate and model the unknown feasible region (Ω) within the search space (i.e., "feasibility determination" problem), without any assumption on the number of constraints as well as their independence. In [44] a Probabilistic SVM (PSVM) is used to calculate the "probability of feasibility" and the optimization scheme alternates between (*i*) a global search for the optimal solution, depending on both this probability and the estimated value of the objective function—modelled through a GP—and (*ii*) a local refinement of the PSVM through an adaptive local sampling scheme. A similar approach was proposed in [31] in which a Least-Square SVM (LS-SVM) is used instead of a PSVM. The output of the LS-SVM is used, as for PSVM, to provide a "probability of feasibility".

The proposed approach, namely SVM-CBO (SVM Constrained BO), instead, is organized in two phases, consecutive and not alternating: the first is aimed to provide a first estimate of Ω (i.e., solving the feasibility determination problem) and the second is "vanilla" BO [1] performed on such an estimate, only. The motivation is that in SVM-CBO feasibility determination is a goal by itself, aimed to obtain a good approximation of the overall feasible region and not only close to the optimal solution. Another relevant difference is that SVM-CBO uses more efficiently the available "budget" (i.e., maximum number of function evaluations): at every iteration, of both phase 1 and 2, just one function evaluation is performed, while in the "boundary refinement" of [44] a given number n_p of function evaluations (namely, auxiliary samples) is performed in the neighborhood of the next point to evaluate (namely, update region), with the aim to locally refine the boundary estimate.

The importance of feasibility determination has been specifically highlighted in real-life problems, such as Pump Scheduling Optimization (PSO) in water distribution systems [45, 46]. While [45] presents an adaptive random search approach to approximate feasible region while optimizing the objective function, [46] proposes a BO framework where infeasibility is treated by assigning a fixed penalty as value of the objective function. As reported in [47], penalty can be also assigned directly to the acquisition function, with the aim to quickly move away from infeasible regions (we refer as "BO with penalty").

The SVM-CBO approach was initially validated on a set of simple 2D test functions [48]; this paper presents a wider set of experiments on test functions generated through the Emmental-type GKLS generator [49], with increasing dimensionality and complexity. The aim was to analyze the effectiveness and efficiency of SVM-CBO with respect to the complexity of the test problem to solve and compare the proposed method to a BO with penalty approach.

The rest of the paper is organized as follows: Sect. 2 provides the background about SVM classification and SMBO, Sect. 3 describes SVM-CBO, Sect. 4 summarizes the experimental setting and Sect. 5 presents the results.

2 Background

2.1 Support Vector Machine classification

The basic idea of Support Vector Machine (SVM) classification [42, 43] consists in searching for a hyper-plane to optimally separate instances (represented as vectors) belonging to two different classes while maximizing the distance of every instance from the hyper-plane (i.e., margin maximization). This first formulation is known as *hard margin SVM* and provides a separation of instances in the two classes without any classification error only in the case that the instances are linearly separable.

Let denote with $D_{i=1:n} = \left(x_i, y_i\right)$ a generic dataset, where $x_i \in \mathbb{R}^d$ is an instance in the d-dimensional space and $y_i \in \{-1, +1\}$ is the "label" of the class it belongs to. Moreover, denote with $w \cdot x - b = 0$ a generic hyper-plane, where $\cdot$ is the scalar product operator. To achieve the margin maximization objective two hyper-planes must be considered: $w \cdot x - b = 1$ and $w \cdot x - b = -1$, where the quantity $2/\|w\|$ is the size of the margin, that is the distance between these two hyper-planes. An important constraint is that no instances must lie in the region between the two hyper-planes, that is $y_i(w \cdot x_i - b) \geq 1$, with $i = 1, \dots, n$. Thus, the hard margin SVM formulation consists in solving the following optimization problem:

$$\begin{aligned} &\min_{w,b} \frac{\|w\|^2}{2} \\ &s.t. \\ &y_i(w \cdot x_i - b) \geq 1 \end{aligned} \tag{1}$$

Given the solution of this problem, the estimated class label is given by $\tilde{y} = \mathrm{sgn}(w \cdot x - b)$, namely the *hyperplane decision function*. The optimization problem is dealt with by introducing Lagrangian relaxation, where the constraints are incorporated into the new objective function:

$$\min_{w,b,\alpha} L(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n} \alpha_i \big(y_i(x_i \cdot w + b) - 1\big) \tag{2}$$

with α_i named Lagrangian coefficients.

If the i-th constraint is violated, then L can be increased by increasing the corresponding α_i. At the same time, w and b will have to change such that L decreases. On the contrary, if the i-th constraint is not met as equality, the relative α_i must be 0: this is the value of α_i that maximizes L (according to Karush–Kuhn–Tucker complementary conditions of optimization theory).

The identification of a saddle point of L is the solution of the Lagrangian problem. In a saddle point the derivatives of L with respect to the original decision variables w and b will vanish, that is $\frac{\partial}{\partial w}L(w, b, \alpha) = 0$ and $\frac{\partial}{\partial b}L(w, b, \alpha) = 0$, leading to $\sum_{i=1}^{n} \alpha_i y_i = 0$ and $w = \sum_{i=1}^{n} \alpha_i y_i x_i$. Thus, the solution vector w is an expansion in terms of a subset of the training examples, more precisely those whose $\alpha_i > 0$, which are called *support vectors*.

By replacing to $\sum_{i=1}^{n} \alpha_i y_i = 0$ and $w = \sum_{i=1}^{n} \alpha_i y_i x_i$ into the Lagrangian function, the original decision variables w and b are eliminated, leading to the dual quadratic optimization problem usually solved in practice:

$$\begin{aligned} &\max_{\alpha \in \mathbb{R}^n} W(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j) \\ &s.t. \\ &\alpha_i \geq 0 \quad \forall i = 1, \dots, n \\ &\sum_{i=1}^{n} \alpha_i y_i = 0 \end{aligned} \tag{3}$$

Using the equation $w = \sum_{i=1}^{n} \alpha_i y_i x_i$, the hyperplane decision function can be rewritten as follows:

$$\tilde{y} = \mathrm{sgn}\left(\sum_{i=1}^{n} y_i \alpha_i (x \cdot x_i) + b \right) \tag{4}$$

with b computed by using the following equation from KKT conditions:

$$\alpha_i\left[y_i(w\cdot x+b)-1\right]=0 \quad \forall\, i=1,\dots,n \tag{5}$$

meaning that all the support vectors lie on the margin while all the other training examples are irrelevant for the construction of the separation hyperplane and the relative hyperplane decision function.

As already mentioned, the hard margin formulation works only in the case of linearly separable data, a quite rare situation in real life. Therefore, the *soft margin SVM* formulation has been proposed: it relaxes the "perfect" separation constraint by admitting some classification errors and limiting them through a penalization term in the objective function. This formulation of the soft margin SVM classification is also known as C-SVM, where C is a regularization parameter to manage the trade-off between minimization of classification error and maximization of margin:

$$\begin{aligned}&\min_{w,b,\xi}\frac{\|w\|^2}{2}+C\sum_{i=1}^{n}\xi_i\\&s.t.\\&y_i\left(w\cdot x_i-b\right)\geq 1-\xi_i\\&\xi_i\geq 0\end{aligned} \tag{6}$$

Nevertheless, both hard and soft margin work by using a linear hyper-plane to separate the two classes of instances, a still strong precondition in real-world classification problems. Thus, the so-called *kernel trick* has been proposed to perform—implicitly—a mapping of the instances from the original space (Input Space) in a new one (Feature Space) where they could be hopefully linearly separated.

Let denote with $x\in\mathbb{R}^d$ a training example in the Input Space and with $\Phi(x)\in\mathbb{R}^p$ its representation in the Feature Space—usually $p>d$, then $k\left(x,x'\right)=\Phi(x)\cdot\Phi\left(x'\right)$—namely the kernel—is the dot product between two training points computed in the Feature Space induced by $\Phi(\,)$. The identification of the $\Phi(\,)$ allowing for linear separation, in the Feature Space, of a set of training examples which are not linearly separable in the Input Space, is an NP-hard problem. The kernel allows for computing similarity between two points in the induced Feature Space without explicitly computing $\Phi(\,)$. Several types of kernel, for SVM classification, have been proposed, such as e.g. Polynomial, Radial Basis Functions (RBF), Sigmoid, etc.

Independently on the type of kernel, the hyperplane decision function can be rewritten as follows:

$$\tilde{y}=\operatorname{sgn}\left(\sum_{i=1}^{n}y_i\alpha_i\left(\Phi(x)\cdot\Phi\left(x_i\right)\right)+b\right)=\operatorname{sgn}\left(\sum_{i=1}^{n}y_i\alpha_i k\left(x,x_i\right)+b\right) \tag{7}$$

and the quadratic optimization problem can be reformulated as follows:

$$\begin{aligned}
&\max_{\alpha\in\mathbb{R}^n} W(\alpha) = \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j k\left(x_i, x_j\right)\\
&s.t.\\
&\alpha_i \geq 0 \quad \forall\, i = 1,\dots,n\\
&\sum_{i=1}^{n}\alpha_i y_i = 0
\end{aligned} \tag{8}$$

Finally, the Gaussian RBF kernel is used in this paper, which is defined as:

$$k\left(x, x'\right) = \exp\left(-\frac{\|x - x'\|^2}{2\sigma_{svm}^2}\right) \tag{9}$$

where σ_{svm}^2 sets the similarity of the two points, x and x' in the Feature Space. The larger the value of σ_{svm}^2 the more similar the two points are considered. Since the kernel concept also arises in sequential model based optimization, σ_{svm}^2 is here used to denote the hyper-parameter of the Gaussian RBF kernel for SVM classifier, while ℓ is used to denote the hyper-parameters in the Squared Exponential (SE) kernel introduced in the following section. As better explained in the following, the mathematical formulation of the two kernel is practically the same, but the role of the two kernels is completely different.

2.2 Sequential model based optimization

The goal of sequential model based optimization (SMBO) is to find a global minimizer of the black-box, multi-extremal and expensive objective function $f(x)$:

$$x^* = \text{argmin}_{x\in X} f(x) \tag{10}$$

where $X \subset \mathbb{R}^d$. In GO, X is usually a bounded box search space.

The SMBO framework consists of 2 main components:

- The first is a (probabilistic) model of the objective function (also known as surrogate model), sequentially updated depending on observations of the objective function (i.e., function evaluations). When a probabilistic model is used, as in Bayesian Optimization (BO), both mean and variance of the surrogate model are updated. A typical choice of probabilistic surrogate model is a Gaussian Process (GP) [50].
- The second key component is an acquisition function, defined on the (probabilistic) surrogate model, whose optimization drives the querying process by identifying the next most "promising" point where the objective function is to be evaluated.

Some basics on GP and a presentation on a set of widely adopted acquisition functions are reported in the following sub-sections.

2.3 The probabilistic model

A GP is an extension of the multivariate Gaussian distribution to an infinite dimensions stochastic process. Just as a Gaussian distribution is a distribution over a random variable, completely specified by its mean and covariance, a GP is a distribution over functions, completely specified by its mean function $\mu(x)$ and covariance function, also known as kernel, $k(x, x')$:

$$f(x) \sim GP(\mu(x); k(x, \mathrm{x}')) \tag{11}$$

It is often useful to intuitively think of a GP as analogous to a function, but instead of returning a single numeric value $f(\bar{x})$ for an arbitrary $\bar{x}$, it returns the mean and variance of a normal distribution over the possible values of $f(\bar{x})$.

In SMBO, a GP is used as a probabilistic surrogate model by "training" according to the current set of function evaluations. Let consider to be at iteration t, that means t evaluations of the objective function have been already performed and stored in the dataset $D_{1:t} = (x_i, f(x_i))$. One of the most important property of GP is that a GP is a collection of random variables, any finite number of which have a joint Gaussian distribution, so that $f(x_{t+1})$ can be obtained as follows. For the sake of simplicity, we use f_t as short notation for $f(x_t)$:

$$\begin{bmatrix} \{f_1, \dots, f_t\} \\ f_{t+1} \end{bmatrix} \sim N\left(0, \begin{bmatrix} \mathrm{K} & \mathrm{k} \\ \mathrm{k}^T & k(x_{t+1}, x_{t+1}) \end{bmatrix}\right) \tag{12}$$

where

$$\mathrm{k} = [k(x_{t+1}, x_1) k(x_{t+1}, x_2) \dots k(x_{t+1}, x_t)] \tag{13}$$

It is then easy to derive an expression for the predictive distribution:

$$P(f_{t+1} | D_{1:t}, x_{t+1}) = \mathcal{N}(f_{t+1} | \mu_t(x_{t+1}), \sigma_t^2(x_{t+1})) \tag{14}$$

where

$$\mu_t(x_{t+1}) = \mathrm{k}^T [\mathrm{K} + \lambda_t^2 I]^{-1} \{f_1, \dots, f_t\} \tag{15}$$

$$\sigma_t^2 = k(x_{t+1}, x_{t+1}) - \mathrm{k}^T [\mathrm{K} + \lambda_t^2 I]^{-1} \mathrm{k} \tag{16}$$

and λ_t^2 is the variance of noise. The function values are drawn according to a multivariate normal distribution $\mathcal{N}(0, \mathrm{K})$, where the kernel matrix K is given by:

$$\mathrm{K} = \begin{bmatrix} k(x_1, x_1) & \dots & k(x_1, x_t) \\ \vdots & \ddots & \vdots \\ k(x_t, x_1) & \dots & k(x_t, x_t) \end{bmatrix} \tag{17}$$

Many covariance functions have been proposed in literature, such as Squared Exponential, Exponential, Matérn, Rational Quadratic, etc. [50]. One of the most widely adopted is the Squared Exponential (SE) kernel, that is the:

$$k(x, x') = \exp\left(-\frac{1}{2\ell^2} \|x - x\|'^2\right) \tag{18}$$

where ℓ is named *characteristic lengthscale*. The SE kernel approaches to 1 as the distance between x and x' decreases to 0, while approaches to 0 when the two points are distant. This means that two points that are close together can be expected to have a very large mutual influence, whereas distant points have almost none.

Although SMBO, especially BO, has become a relevant approach in the case there is not any information about derivatives, when gradients of $f(x)$ are available or can be inferred they can be incorporated, for instance into the GP, to improve the optimization process via local search [51].

2.4 The acquisition function

The acquisition function is the mechanism to implement the trade-off between exploration and exploitation in the SMBO. The basic idea is to improve over the best solution observed so far ("best seen"), even if an improvement is not guaranteed from one iteration to the next, due to the exploration. Any acquisition function aims to guide the search of the optimum towards points with potentially low values of objective function (in minimization problem) either because the prediction of $f(x)$, based on the surrogate model, is high or the uncertainty is high (or both). While *exploiting* means to consider the area providing more chance to improve the current solution (with respect to the current surrogate model), *exploring* means to move towards less explored regions of the search space. Many acquisition functions have been proposed, such as Probability of Improvement, Expected Improvement, Confidence Bound (Upper/Lower Confidence Bound for maximization/minimization problems, respectively), Entropy Search and Predictive Entropy Search and Knowledge Gradient, among the most well-known [1, 3, 52].

3 Support Vector Machine constrained bayesian optimization

3.1 Problem formulation

We start with the definition of the problem, that is:

$$\min_{x\in\Omega\subset X\subset\mathbb{R}^d} f(x) \tag{19}$$

where $f(x)$ has the following properties: it is black-box, multi-extremal, expensive and partially defined. Moreover, we consider the case that constraints defining the feasible region are also black-box. This can be a typical setting in optimization problems where the objective function is computed by a, usually time consuming, simulation process/software for each $x \in X$ [53]. Following, some useful notation:

- x_i is the i-th point evaluated;
- $y_i = \{+1, -1\}$ defines if x_i is feasible or infeasible, respectively;
- $D_n^\Omega = \left\{\left(x_i, y_i\right)\right\}_{i=1,\dots,n}$ is the **feasibility determination** dataset;
- $D_l^f = \left\{\left(x_i, f\left(x_i\right)\right)\right\}_{i=1,\dots,l}$ is the **function evaluations** dataset, with l the number of feasible points out of the n evaluated so far:

$$l = \left|(x, y) \in D_n^\Omega\, :\, y = 1\right| \tag{20}$$

Thus, it is easy to notice that the feasibility determination dataset, D_n^Ω, is exactly in the form of any generic dataset which SVM classification can be applied on.

3.2 Phase 1: feasibility determination

The first phase of SVM-CBO aims at finding an estimate $\tilde{\Omega}$ of the actual feasible region Ω, in M function evaluations. Feasibility of any point $x \in X$ is estimated depending on the (non-linear) separation hypersurface induced by an SVM classifier trained on D_n^{Ω}. The SVM classifier uses an RBF kernel to model a non-linear boundary for the feasible region. Let denote with $h_n(x)$ the argument of the SVM-based classification function:

$$h_n(x) = \sum_{i=1}^{n_{SV}} \alpha_i y_i k\left(x_i, x\right) + b \tag{21}$$

where α_i and y_i are the Lagrangian coefficient and the "feasibility label" of the *i*th support vector respectively, $k(.,.)$ is the SVM kernel function (i.e., an RBF kernel, in this study), b is the offset and n_{SV} is the number of support vectors.

The boundary of the estimated feasible region $\tilde{\Omega}_n$ is given by $h_n(x) = 0$. Finally, the estimated feasibility for a given point $x \in X$ is:

$$\tilde{y} = \operatorname{sign}\left(h_n(x)\right) = \begin{cases} +1 & \textit{if } x \in \tilde{\Omega}_n \\ -1 & \textit{if } x \notin \tilde{\Omega}_n \end{cases} \tag{22}$$

In phase 1 of SVM-CBO, the next promising point is selected according to two different goals:

- Improving the estimate of feasible region
- Discovering possible disconnected feasible regions

To deal with the first goal, the distance from the currently estimated boundary of $\tilde{\Omega}_n$ is used:

$$d_n\left(h_n(x), x\right) = \left|h_n(x)\right| = \left|\sum_{i=1}^{n_{SV}} \alpha_i y_i k\left(x_i, x\right) + b\right| \tag{23}$$

With respect to the second goal, a measure of "coverage of the search space" is used:

$$c_n(x) = \sum_{i=1}^{n} e^{-\frac{\|x_i - x\|^2}{2\sigma_c^2}} \tag{24}$$

So, $c_n(x)$ is a sum of n RBF functions centred on the points evaluated so far, with σ_c a parameter to set the width of the corresponding bell-shaped curve.

Finally, the next point is selected by solving the following optimization problem:

$$x_{n+1} = \underset{x \in \mathrm{X}}{\operatorname{argmin}} \left\{ d_n\left(h_n(x), x\right) + c_n(x) \right\} \tag{25}$$

Thus, one wants to select the point associated to minimal coverage (i.e., max uncertainty) and minimal distance from the boundaries of the current estimated feasible region. This allows to balance between improving the estimate of the feasible region and discovering possible disconnected feasible sub-regions (in less explored areas of the search space). It is important to highlight that, in phase 1, the point solving this optimization problem is searched for on the overall bounded-box search space X.

At each iteration, the function is evaluated at the new point x_{n+1} and the collected information is used to update datasets and the SVM classification model. The process iterates until M function evaluations are reached. The algorithm for SVM-CBO phase 1 (i.e., feasibility determination) is summarized.

PHASE 1 ALGORITHM: Feasibility Determination

```
INPUT:
  $N$, overall number of function evaluations (i.e., budget)
  $M$, number of function evaluations in phase 1 (included in $N$)
  $p$, number of initial randomly sampled points (included in $M$)
INITIALIZATION:
  $D_p^\Omega = \{(x_i, y_i)\}_{i=1,\dots,p}$   # feasibility determination dataset
  $D_l^f = \{(x_i, f(x_i))\}_{i=1,\dots,l}$   # function evaluations dataset, with $l$ as defined in (Eq. 20)
MAIN:
  $n \leftarrow p$
  while $n < M$ do
    $h_n(x) \leftarrow train\ an\ SVM\ classifier\ on\ D_n^\Omega$
    $x_{n+1} \leftarrow \underset{x \in X}{\operatorname{argmin}}\{d_n(h_n(x), c_n(x))\}$
    evaluate $y_{n+1}, f(x_{n+1})$
    $D_{n+1}^\Omega \leftarrow D_n^\Omega \cup \{(x_{n+1}, y_{n+1})\}$
    if $y_{n+1} = +1$ then
      $D_{l+1}^f \leftarrow D_l^f \cup \{(x_{n+1}, f(x_{n+1}))\}$
      $l \leftarrow l + 1$
    endif
    $n \leftarrow n + 1$
  endwhile
OUTPUT:
  $D_M^\Omega, D_M^f, h_M(x)$ and $\tilde{\Omega}_M = \{x \in X : h_M(x) > 0\}$
```

3.3 Phase 2: BO constrained to the estimated feasible region

Phase 2 of SVM-CBO consists of a "vanilla" BO process but modified as follows:

- the search space (related to this phase) is not the overall box-bounded domain X but constrained to the estimated feasible region $\tilde{\Omega}_n$ identified in phase 1
- a GP is fitted to provide a probabilistic surrogate model of the objective function, but it is fitted by only using the feasible observations stored so far into D_l^f
- Lower Confidence Bound (LCB) is used as acquisition function in this phase, but it is defined on $\tilde{\Omega}_n$, only. Thus, the next point to evaluate is given by:

$$x_{n+1} = \underset{x \in \tilde{\Omega}_n}{\operatorname{argmin}} \left\{ LCB_n(x) = \mu_n(x) - \beta_n \sigma_n(x) \right\} \tag{26}$$

where $\mu_n(x)$ and $\sigma_n(x)$ are the mean and the standard deviation of the current GP and β_n is the inflate parameter to deal with the trade-off between exploration and exploitation for this phase. Theoretically motivated guidelines for setting and scheduling β_n to achieve optimal regret are reported in [54]. It is important to highlight that, contrary to phase 1, the acquisition function is here minimized on $\tilde{\Omega}_n$, only, instead of the entire bounded-box search domain X.

The point x_{n+1} is just expected to be feasible, according to $\tilde{\Omega}_n$, but the information on its actual feasibility is known only after evaluating $f\left(x_{n+1}\right)$. The collected information is used to update datasets, GP and—if needed—the SVM classification model. The process iterates until an overall number of N function evaluations, including the M already performed in phase 1, is reached. The algorithm for SVM-CBO phase 2 (i.e., BO constrained to the estimated feasible region) is summarized.

PHASE 2 ALGORITHM: BO constrained to the estimated feasible region

```
INPUT:
  $N$, $M$ and $p$, as defined in phase 1
  $D_M^\Omega$, $D_M^f$, $h_M(x)$ and $\tilde{\Omega}_M$, resulting from phase 1
MAIN:
  $n \leftarrow M$
  while $n < N$ do
    $(\mu_n(x), \sigma_n(x)) \leftarrow$ train a GP on $D_l^f$, with $l$ defined as in (Eq. 20)
    $x_{n+1} \leftarrow \underset{x \in X}{\operatorname{argmin}}\{\mu_n(x) + \beta_n \sigma_n(x)\}$
    evaluate $y_{n+1}, f(x_{n+1})$
    $D_{n+1}^\Omega \leftarrow D_n^\Omega \cup \{(x_{n+1}, y_{n+1})\}$
    if $y_{n+1} = +1$ then
      $D_{l+1}^f \leftarrow D_l^f \cup \{(x_{n+1}, f(x_{n+1}))\}$
      $h_{n+1}(x) \leftarrow h_n(x)$ # there is no need to retrain the SVM classifier
      $l \leftarrow l + 1$
    else
      $h_{n+1}(x) \leftarrow$ train an SVM classifier on $D_{n+1}^\Omega$
    endifelse
    $n \leftarrow n + 1$
  endwhile
OUTPUT:
  $D_M^\Omega$, $D_M^f$, $h_M(x)$ and $\tilde{\Omega}_M = \{x \in X : h_M(x) > 0\}$
```

The following figure shows three different iterations of SVM-CBO: grey area is the infeasible region $X \backslash \Omega$, light-blue area is its estimate $X \backslash \tilde{\Omega}_n$, the blue dashed line is the estimated boundary, crosses and blue points are, respectively, infeasible and feasible evaluations and the red star is the unique feasible global minimizer (Fig. 1). At the end of initial LHS sampling (left) the approximation of the feasible region is quite poor, while at the end of phase 1 (middle) it is good, with both the two disconnected feasible subregions modelled. Although points in phase 1 are selected with the only aim to improve the estimate of the feasible region, there are also 2 points which have been sampled closely to the global minimizer. This is due to the proximity of the global minimizer to the feasible region's boundary (middle). All the points selected in phase 2 are significantly close to the global minimizer (right), without requiring any further refinement of the feasible region estimate.

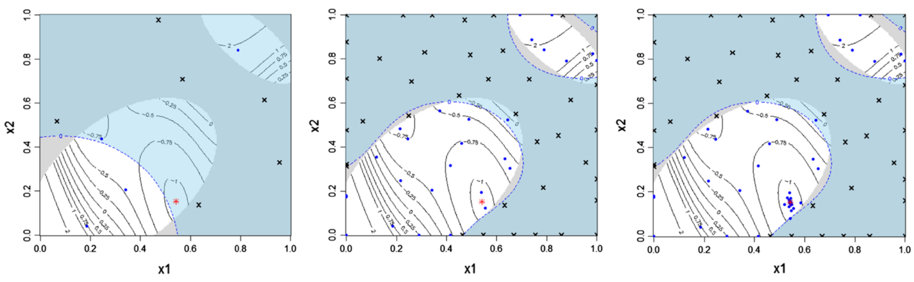


Fig. 1 Estimates of the feasible region for the "Brainin constrained to two disconnected ellipses" test function (presented in Sect. 4.1). Estimate at the end of: (left) initialization, (middle) phase 1 and (right) phase 2. Blue dashed line is the boundary of $\tilde{\Omega}_n$. Crosses and blue points are infeasible and feasible points, respectively; in red the unique feasible global minimizer. (Color figure online)

It is possible to notice some errors in the SVM classification: one blue (feasible) point is outside the estimated feasible region, while two crosses (infeasible points) are inside. However, it is not important to have a perfect estimate of the feasible region, a good approximation, just like the one in figure, is enough.

4 Experiments

4.1 Simple 2D test functions

The SVM-CBO approach has been validated on three well-known 2D test functions for CGO, that are: Rosenbrock constrained to a disk [55], Rosenbrock constrained to a line and a cubic [55, 56], and Mishra's Bird constrained [57]. Since these test functions are all constrained to just one connected feasible region, two more supplementary test functions have been defined: Branin (rescaled) [58] constrained to an ellipse and Branin (rescaled) constrained on two disconnected ellipses. These two further tests allowed us to validate SVM-CBO with respect to a connected or disconnected feasible region, on the same objective function. Analytical expressions of these test functions are in [48].

4.2 Test functions generator: Emmental-type GKLS

Emmental-type GKLS [49] is a software for generating multidimensional continuously differentiable multiextremal objective functions and non-linear constraints. The global minimizer of these constrained test functions is always located on the boundary of the feasible region. Dimensionality and complexity of the generated test functions can be easily controlled by setting few generator's parameters.

The experimental setting defined for the experiments with Emmental-type GKLS was inspired by the one used in [5]: test functions of growing dimensionality, from 2D to 5D, have been considered, for 2 different classes of complexity each. In this paper, 10 different functions for each dimensionality and class of complexity have been generated, leading to $4 \times 2 \times 10 = 80$ test functions. The number of local minima is 30, the number of constraints is 20, 2 and 10, respectively for the possible three types considered by Emmental-type GKLS, with 5 active constraints at the global optimum.

4.3 Experimental setting

SVM-CBO was validated on a budget of 100 function evaluations, divided in:

- 10 for initialization, through uniform random sampling;
- 60 for phase 1 (i.e., feasibility determination)
- and 30 for phase 2 (i.e., BO constrained to the estimated feasible region)

This budget is clearly insufficient to solve the CGO test functions, thus SVM-CBO, and BO in general, cannot guarantee an optimal solution differing less than a given small ε from the optimum value function [5]. On the other hand, it is important to highlight that the aim of the proposed approach is to obtain a good solution under very limited budget, and that this solution is better than the one provided by BO assigning a fixed penalty to infeasible evaluations. Thus, the goal of the experiments was to validate whether SVM-CBO is more effective, and efficient, than "BO with penalty" in the case of a partially defined $f(x)$ (a.k.a. incomputable domains, crush constraints, etc.).

In the case of BO with penalty, the overall budget was the same but divided as follows:

- 10 evaluations for initialization through uniform random sampling;
- and 90 for the BO process (the sum of phase 1 and phase 2 budgets in SVM-CBO).

It is important to highlight that, for each independent run, the initial set of solutions identified through uniform random sampling was the same for both SVM-CBO and BO with penalty, avoiding differences in the results due to different initializations.

SVM-CBO was analyzed with respect to different aspects:

- does the SVM-CBO offer a better solution than BO with penalty? Or, at least, the same solution but with less function evaluations?
- is the number of test problems solved by SVM-CBO higher than BO with penalty?

To address the first analysis, *Gap metric* [59, 60] was considered, which measures the improvement obtained along a sequential optimization process with respect to global optimum $f(x^*)$ and the initial best value $f\left(x_0\right)$ observed during initialization:

$$G_n = \frac{\left|f\left(x^+\right) - f\left(x_0\right)\right|}{\left|f(x^*) - f\left(x_0\right)\right|}$$

where $f\left(x^+\right)$ is the "best seen" up to iteration n. Gap metrics varies in the range [0, 1].

For statistical significance, the Gap metric was computed on 30 different runs, for every test function and for both SVM-CBO and BO with penalty.

To address the second analysis, the *Operating Zones* methodology was considered [5], an extension of the *Operational Characteristics* [61] to the performance analysis of stochastic global optimization methods. The Operating Zones ranges in 0 and the number of problems to solve or can be reported in percentage terms.

5 Results

5.1 Results on simple 2D test functions

As already reported in [48], SVM-CBO outperforms BO with penalty on all the five 2D test functions considered. At the end of the optimization process, BO with penalty provided solutions with a gap metric—averaged on 30 runs for every function—ranging in 0.55–0.70, while SVM-CBO provided solutions with gap metric not lower than 0.80. Furthermore, gap metric variance was always lower in the case of SVM-CBO, at the end of the optimization process. For all the 2D test functions the behavior was similar: SVM-CBO's gap metric drastically improves when the phase 2 starts. The following figure shows two exemplary cases: Rosenbrock constrained to a cubic and a line (left) and Branin constrained to two disconnected ellipse (right).

Although the acquisition function in phase 1 is aimed at sampling points to improve the estimate of feasible region, in some cases—just like in Fig. 2 (right)—it can also offer a better gap metric with respect to BO with penalty, also in phase 1. Finally, results on Rosenbrock constrained to a disk are like those in Fig. 2 (left), while for Branin constrained on an ellipse and Mishra's function results are like those in Fig. 2 (right).

5.2 Results on Emmetal-type GKLS test functions

Due to the large number of generated test functions (i.e., 80), and 30 different runs each, for both SVM-CBO and BO with penalty, all the results have been summarized in the following 4 tables, where each table refers to a dimensionality, from 2D to 5D (Tables 1, 2, 3, 4).

The definition of "simple" and "hard" classes follows from [5]; the column "seed" specifies the value used to generate the specific test function for any dimensionality-class pair. Tables report the best seen at end of the optimization process (mean and standard deviation on 30 independent runs), separately for SVM-CBO and BO with penalty. The value of the Mann–Withney's U test is reported along with the associated p value. Numbers in bold indicate significantly better results: symbols *, ** and *** indicate a p value less or equal than 0.05, 0.01 and 0.001, respectively.

BO with penalty never outperforms SVM-CBO. Moreover, the higher the dimensionality—and harder is the class—of the problem, the higher the significance of the difference between the best seen obtained by SVM-CBO and BO with penalty.

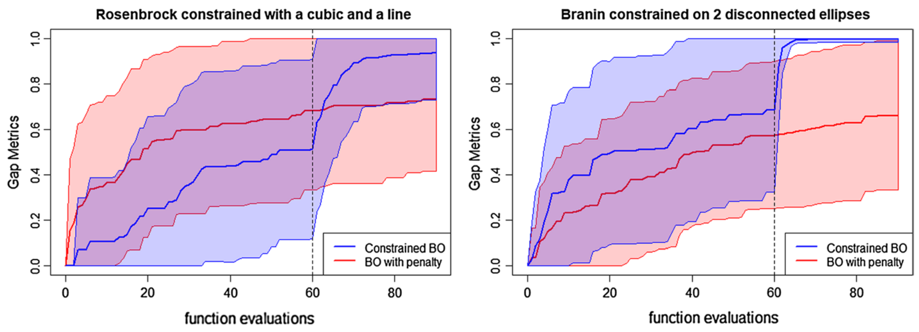


Fig. 2 Comparison of Gap metrics for SVM-CBO (namely, Constrained BO in the legend) and BO with penalty on two out the five simple 2D test functions

Table 1 2D Emmental-type GKLS functions: best seen for SVM-CBO and BO with penalty

Class	Seed	SVM-CBO	BO penalty	–Withney	*p* value	Sign. level
Simple	1	**– 0.50 ± 0.14**	– 0.25 ± 0.18	52.50	< 0.001	***
	2	**– 0.07 ± 0.09**	0.07 ± 0.08	52.00	< 0.001	***
	3	**0.09 ± 0.03**	0.12 ± 0.08	194.50	0.892	
	4	0.18 ± 0.01	0.17 ± 0.09	149.00	0.174	
	5	– 0.08 ± 0.17	– 0.11 ± 0.21	160.00	0.285	
	6	**– 0.13 ± 0.09**	– 0.05 ± 0.07	98.00	0.005	**
	7	0.03 ± 0.06	0.03 ± 0.03	189.00	0.776	
	8	– 0.18 ± 0.15	– 0.22 ± 0.16	189.50	0.787	
	9	**0.01 ± 0.00**	0.02 ± 0.02	103.00	0.009	**
	10	– 0.02 ± 0.06	– 0.05 ± 0.16	155.50	0.234	
Hard	1	**– 0.50 ± 0.15**	– 0.29 ± 0.2	64.00	< 0.001	***
	2	0.08 ± 0.02	0.08 ± 0.04	209.50	0.808	
	3	**0.07 ± 0.01**	0.10 ± 0.05	75.50	0.001	***
	4	0.18 ± 0.01	0.17 ± 0.06	159.50	0.279	
	5	**– 0.31 ± 0.07**	– 0.24 ± 0.05	20.00	< 0.001	***
	6	**0.02 ± 0.02**	0.06 ± 0.06	112.50	0.019	*
	7	**– 0.04 ± 0.07**	– 0.01 ± 0.07	114.50	0.021	*
	8	0.05 ± 0.01	0.05 ± 0.07	149.00	0.174	
	9	0.01 ± 0.01	0.02 ± 0.03	127.50	0.051	
	10	**0.01 ± 0.01**	0.03 ± 0.03	57.50	< 0.001	***

With increasing dimensionality, both SVM-CBO and BO with penalty have less chances to reach the attraction basin of the global minimizer. This is due to the very narrow basin of attraction, the complicated location of the optimizer (on the boundary of the feasible region) and to the limited budget (only 100 function evaluations, overall). However, BO is usually adopted in practical problems just to find a solution with a good function value, under a limited budget, rather than the global minimizer. Experiments were devoted to validating if the information about feasibility/infeasibility can be effectively exploited to improve effectiveness and efficiency with respect to BO approaches using a simple fixed penalty. Extended results presented in this paper prove that, in the case of a partially defined objective function and black-box constraint, the simple adoption of a penalty value is completely misleading.

To compute the Operating Zones the following process has been used: for every test function the value of the best seen at the end of the optimization, separately for SVM-CBO and BO with penalty, was averaged on the 30 different runs, namely $f^{+(i)}_{SVMCBO}$ and $f^{+(i)}_{BO}$, where the apex (i) refers to the i th test function (dimensionality, class and seed). Then, the minimum between these two values was selected for each test function, that is $\bar{f}^{(i)} = \min\left\{f^{+(i)}_{SVMCBO}, f^{+(i)}_{BO}\right\}$. The value $\bar{f}^{(i)}$ was used as threshold, for both SVM-CBO and BO with penalty, to consider the i th test function as solved, in each one of the 30 independent runs. In this way, every test problem is considered as solved at least once. The main motivation for this selection of the thresholds $\bar{f}^{(i)}$ is that neither SVM-CBO nor BO with penalty can guarantee an optimal solution differing less than a given small ε from the optimum value function.

Table 2 3D Emmental-type GKLS functions: best seen for SVM-CBO and BO with penalty

Class	Seed	SVM-CBO	BO penalty	–Withney	*p* value	Sign. level
Simple	1	**0.04 ± 0.02**	0.13 ± 0.21	44.00	< 0.001	***
	2	**– 0.35 ± 0.16**	– 0.04 ± 0.13	25.00	< 0.001	***
	3	**0.09 ± 0.07**	0.20 ± 0.18	89.00	0.003	**
	4	**0.03 ± 0.04**	0.12 ± 0.06	42.00	< 0.001	***
	5	**0.05 ± 0.04**	0.12 ± 0.08	82.00	0.001	**
	6	**0.07 ± 0.06**	0.16 ± 0.01	88.50	0.003	**
	7	**0.13 ± 0.04**	0.19 ± 0.13	118.00	0.027	*
	8	**0.06 ± 0.01**	0.12 ± 0.05	29.00	< 0.001	***
	9	**– 0.01 ± 0.06**	0.07 ± 0.05	48.00	< 0.001	***
	10	**0.14 ± 0.02**	0.19 ± 0.07	99.00	0.006	**
Hard	1	**– 0.03 ± 0.15**	0.14 ± 0.12	39.50	< 0.001	***
	2	**– 0.36 ± 0.11**	– 0.03 ± 0.08	8.00	< 0.001	***
	3	**0.13 ± 0.04**	0.18 ± 0.08	105.00	0.009	**
	4	**0.02 ± 0.04**	0.13 ± 0.11	26.00	< 0.001	***
	5	**0.03 ± 0.02**	0.14 ± 0.09	31.00	< 0.001	***
	6	**0.06 ± 0.05**	0.17 ± 0.08	36.00	< 0.001	***
	7	**0.11 ± 0.03**	0.18 ± 0.08	61.00	< 0.001	***
	8	**0.06 ± 0.02**	0.13 ± 0.07	39.00	< 0.001	***
	9	**0.02 ± 0.04**	0.11 ± 0.11	61.00	< 0.001	***
	10	**0.13 ± 0.03**	0.19 ± 0.06	69.50	< 0.001	***

Figure 3 reports the Operating Zones of SVM-CBO and BO with penalty when (left) test functions are solved in at least 30% of the independent runs and (right) in at least 50% of the independent runs.

Again, the most relevant result is the significant increase obtained by SVM-CBO after the end of phase 1. The Operating Zones confirm that exploiting the information represented by a good approximation of the unknown feasible region can significantly improve the effectiveness of BO, compared to assigning a fixed penalty to infeasible evaluations.

6 Discussion

Finally, this section provides some further discussion, regarding computational costs and convergence results. Starting from computational costs, SVM-CBO results to be more efficient than BO with penalty. In Fig. 4 the value of the best seen over time is reported for both SVM-CBO and BO with penalty for the same test problem (i.e., a 5D hard test function). To exclude any other computational load and make uniform the comparison, the computational time is not the wall-clock time but, at a given iteration $\bar{n}$, it is computed as $\sum_{n=1}^{\bar{n}} O(n^3)$. For visualization purposes it is assumed $O(1^3) = 1$ unit of time. From one function evaluation to the next, the computational time of BO with penalty always increases. The reason is that BO with penalty uses all the points evaluated so far to fit the probabilistic surrogate model. On the contrary, SVM-CBO—in phase 2—uses only feasible points to fit the probabilistic surrogate model of the objective function, and they are

Table 3 4D Emmental-type GKLS functions: best seen for SVM-CBO and BO with penalty

Class	Seed	SVM-CBO	BO penalty	–Withney	*p* value	Sign. level
Simple	1	**0.16 ± 0.04**	0.41 ± 0.15	9.00	< 0.001	***
	2	**0.18 ± 0.06**	0.42 ± 0.15	5.00	< 0.001	***
	3	**0.25 ± 0.02**	0.35 ± 0.08	23.00	< 0.001	***
	4	**0.16 ± 0.02**	0.28 ± 0.12	55.00	< 0.001	***
	5	**0.30 ± 0.04**	0.48 ± 0.26	57.50	< 0.001	***
	6	0.20 ± 0.05	0.23 ± 0.06	135.00	0.081	
	7	**-0.26 ± 0.36**	0.22 ± 0.23	52.50	< 0.001	***
	8	**0.04 ± 0.12**	0.29 ± 0.17	28.50	< 0.001	***
	9	**0.24 ± 0.04**	0.46 ± 0.31	44.50	< 0.001	***
	10	**0.20 ± 0.03**	0.47 ± 0.21	22.00	< 0.001	***
Hard	1	**0.15 ± 0.05**	0.30 ± 0.09	18.50	< 0.001	***
	2	**0.21 ± 0.06**	0.40 ± 0.14	30.50	< 0.001	***
	3	**0.22 ± 0.03**	0.35 ± 0.13	57.50	< 0.001	***
	4	**0.15 ± 0.03**	0.27 ± 0.18	74.50	0.001	***
	5	**0.33 ± 0.02**	0.44 ± 0.15	56.00	< 0.001	***
	6	**0.19 ± 0.03**	0.25 ± 0.06	53.00	< 0.001	***
	7	**-0.25 ± 0.38**	0.25 ± 0.24	52.00	< 0.001	***
	8	**0.10 ± 0.11**	0.33 ± 0.18	35.00	< 0.001	***
	9	**0.23 ± 0.03**	0.41 ± 0.17	36.00	< 0.001	***
	10	**0.21 ± 0.02**	0.41 ± 0.21	18.00	< 0.001	***

usually significantly less than the overall points evaluated so far. In any case the time from one function evaluation to the next, for SVM-CBO, is not monotone because the selection of an infeasible point requires to update both the estimate of the feasible region—that is retraining the SVM classifier with complexity $O(\bar{n}^3)$—and the probabilistic surrogate model of the objective function—that is fitting the GP, also with complexity $O(l^3)$, where l is the number of feasible evaluations, only.

As far as the convergence is concerned, some theoretical results have been already provided in [54] for BO with GP as probabilistic surrogate model and Lower/Upper Confidence Bound as acquisition function, even if in the case of bounded-box search space. In [30], instead, convergence proofs for EI in the CGO are provided. This study does not extend the theoretical results provided in the two quoted papers, rather provides extended empirical results to evaluate the practical advantages offered by the proposed SVM-CBO approach.

A further experiment was devoted to investigating the impact of the budget: a 5D hard test function was selected and 3 different runs were performed, where for each run SVM-CBO and BO with penalty shared the same set of initial points uniformly sampled at random. The available budget for each run was of 1000 function evaluations divided into: 100 for initialization, 600 for SVM-CBO phase 1 and 300 for SVM-CBO phase 2 (i.e., 900 for BO with penalty, after initialization). The proportion between initialization, phase 1 and phase 2 is the same used for previous experiments (i.e., 10–60–30%). There is not any specific reason for this schema: the main motivation is that, according to the preliminary experiments on simple 2D test functions, few function evaluations are sufficient to obtain a

Table 4 5D Emmental-type GKLS functions: best seen for SVM-CBO and BO with penalty

Class	Seed	SVM-CBO	BO penalty	–Withney	*p* value	Sign. level
Simple	1	**0.18 ± 0.04**	0.53 ± 0.25	21.00	< 0.001	***
	2	**0.11 ± 0.07**	0.50 ± 0.25	17.00	< 0.001	***
	3	**0.15 ± 0.03**	0.47 ± 0.26	12.00	< 0.001	***
	4	**0.24 ± 0.02**	0.49 ± 0.21	28.00	< 0.001	***
	5	**0.07 ± 0.11**	0.51 ± 0.28	3.00	< 0.001	***
	6	**0.26 ± 0.05**	0.46 ± 0.17	30.00	< 0.001	***
	7	**0.11 ± 0.05**	0.49 ± 0.22	11.00	< 0.001	***
	8	**0.22 ± 0.03**	0.50 ± 0.21	26.00	< 0.001	***
	9	**0.25 ± 0.04**	0.51 ± 0.22	41.50	< 0.001	***
	10	**0.21 ± 0.03**	0.63 ± 0.25	4.00	< 0.001	***
Hard	1	**0.19 ± 0.07**	0.59 ± 0.27	13.00	< 0.001	***
	2	**0.10 ± 0.06**	0.50 ± 0.19	2.00	< 0.001	***
	3	**0.14 ± 0.02**	0.51 ± 0.27	3.00	< 0.001	***
	4	**0.31 ± 0.05**	0.48 ± 0.17	39.00	< 0.001	***
	5	**0.05 ± 0.13**	0.45 ± 0.24	18.00	< 0.001	***
	6	**0.30 ± 0.07**	0.58 ± 0.29	59.50	< 0.001	***
	7	**0.11 ± 0.05**	0.34 ± 0.14	21.00	< 0.001	***
	8	**0.21 ± 0.04**	0.60 ± 0.31	3.00	< 0.001	***
	9	**0.25 ± 0.02**	0.53 ± 0.22	17.00	< 0.001	***
	10	**0.23 ± 0.06**	0.51 ± 0.21	24.00	< 0.001	***

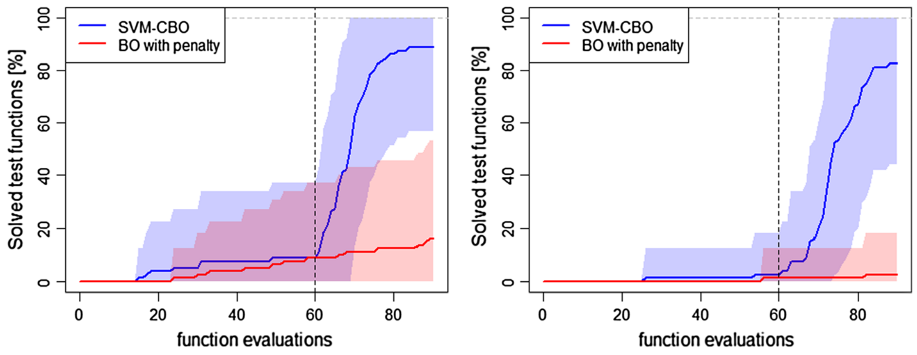
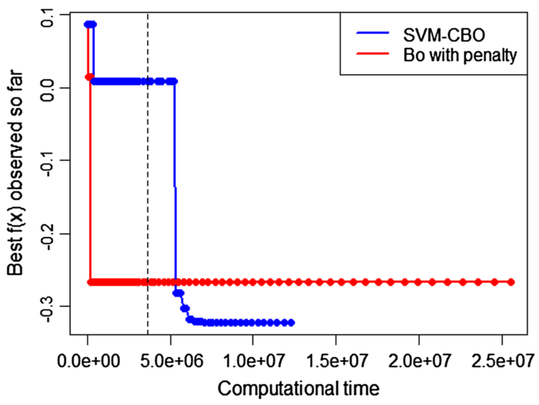


Fig. 3 Operating Zones for SVM-CBO and BO with penalty: percentage of solved problems (mean and standard deviation) in at least 30% (left) and 50% (right) of the 30 independent runs

good solution if the feasible region is well approximated, so most of the budget is allocated to phase 1.

Figure 5 reports the obtained results: SVM-CBO outperformed BO with penalty on all the three runs, and with a relevant difference in terms of best seen on two of them. The main important consideration is that, although starting from the same initial set of points, the strategy used, in phase 1, to select the next point to evaluate is more effective than LCB in BO with penalty. SVM-CBO provides not only an improvement after the start of phase

Fig. 4 Best seen with respect to computational time. To exclude any other computational load and make uniform the comparison, the computational time is not the wall-clock time but, at a given iteration $\bar{n}$, it is computed as $\sum_{n=1}^{\bar{n}} O(^3)$. For visualization it is assumed $O(1^3) = 1$



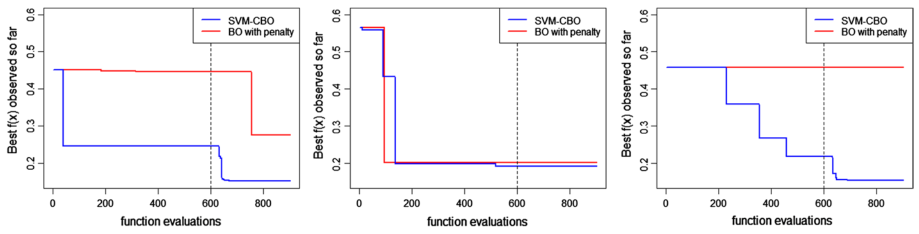


Fig. 5 Gap metrics for SVM-CBO and BO with penalty on three experiments with a larger budget (i.e., 1000 function evaluations, including 100 for initialization)

2, but it can also “optimize” in phase 1—even it is not the aim of that phase—better than BO with penalty. This confirms that using a penalty can dramatically affect both the efficiency (fitting the GP using all the evaluations performed so far) and the effectiveness (best seen at the end of the optimization process) of BO in the case of partially defined objective functions under unknown constraints.

7 Conclusions

The SVM-CBO approach proposed proved to be more effective and computationally efficient than a traditional BO solution which uses a fixed penalty for infeasible function evaluations.

With respect to other constrained BO approaches, SVM-CBO does not require strong assumptions on a priori knowledge about the number of constraints and their independence, even on the objective function. The presented approach considers the computability (aka feasibility) of the objective function as black-box and aims at modelling the boundary of the feasible region overall, instead of “per-constraint”.

With respect to other two recent works which use a classifier to model the entire feasible region, SVM-CBO proposes the following contributions. First, it requires a smaller budget than in [44]: in the quoted reference many function evaluations are required in the neighbourhood of the estimated boundary to improve its approximation. Moreover, this type of "focusing" offers few chances to discover possible disconnected feasible sub-regions (as in Example 1, Fig. 11 of [44]). Second, the organization of SVM-CBO in two consecutive phases allows to preliminary solve the so-called feasibility determination problem—providing useful information about the behaviour of the simulation software or real-life system to be optimized—and then exploit this information for improving effectiveness and efficiency of the optimization process (i.e., phase 2). Moreover, instead of using the "probability of feasibility", which is basically a penalization factor in [44] and [31], SVM-CBO works by constraining the acquisition function, namely LCB, to the current estimate of the feasibility region. This results in a more effective sampling of the new point, as also reported in [44] when probability of feasibility is used to constrain—in probabilistic terms—the acquisition function (i.e., EI) instead of penalize it. Finally, SVM-CBO proposes to use LCB as acquisition function (in phase 2) instead of EI, since LCB is known to offer a more balanced trade-off between exploration and exploitation.

Limitations of SVM-CBO are related to the impossibility to perform any sensitivity analysis with respect to a specific constraint. Since the entire feasible region is modelled by only one SVM classifier, only an analysis of the optimal solution about its robustness with respect "overall feasibility" can be performed. Finally, an important remark is that SVM-CBO assumes that both computable and incomputable (i.e., feasible and infeasible) evaluations can be performed, since computability is a relevant information by itself for the approach. This means that it can be successfully applied to simulation–optimization but not in safe-optimization, where "infeasibility" means destruction of the system to be optimized, a more compelling restriction implying the termination of the optimization process.

References

1. Frazier, P.I.: Bayesian optimization. In: Recent Advances in Optimization and Modeling of Contemporary Problems—INFORMS, pp. 255–278 (2018)
2. Jones, D.R., Schonlau, M., Welch, W.J.: Efficient global optimization of expensive black-box functions. J. Global Optim. **13**(4), 455–492 (1998)
3. Shahriari, B., Swersky, K., Wang, Z., Adams, R.P., De Freitas, N.: Taking the human out of the loop: a review of Bayesian Optimization. Proc. IEEE **104**(1), 148–175 (2016)
4. Žilinskas, A., Žilinskas, J.: Global optimization based on a statistical model and simplicial partitioning. Comput. Math Appl. **44**(7), 957–967 (2002)
5. Sergeyev, Y.D., Kvasov, D.E., Mukhametzhanov, M.S.: On the efficiency of nature-inspired metaheuristics in expensive global optimization with limited budget. Sci Rep-UK **8**(1), 453 (2018)
6. Sergeyev, Y.D., Strongin, R.G., Lera, D.: Introduction to global optimization exploiting space-filling curves. Springer, Berlin (2013)
7. Sergeyev, Y.D., Kvasov, D.E.: Deterministic global optimization: an introduction to the diagonal approach. Springer, Berlin (2017)
8. Zhigljavsky, A., Žilinskas, A.: Stochastic global optimization, vol. 9. Springer, Berlin (2007)
9. Archetti, F., Betrò, B.: A priori analysis of deterministic strategies. Towards Glob. Optim. **2**, 31–48 (1978)
10. Archetti, F., Betrò, B.: Stochastic models and optimization. Bollettino dell'Unione Matematica Italiana **5**(17), 295–301 (1980)
11. Archetti, F., Betrò, B.: A probabilistic algorithm for global optimization. Calcolo **16**, 335–343 (1979)

12. Eggensperger, K., Lindauer, M., Hutter, F.: Pitfalls and best practices in algorithm configuration. J. Artif. Intell. Res. **64**, 861–893 (2019)
13. Hutter, F., Kotthoff, L., Vanschoren, J. (eds.): Automated Machine Learning. Methods, Systems, Challenges. The Springer Series on Challenges in Machine Learning. Springer (2019). https://doi.org/10.1007/978-3-030-05318-5
14. Thornton, C., Hutter, F., Hoos, H.H., Leyton-Brown, K.: Auto-WEKA: combined selection and hyperparameter optimization of classification algorithms. In: Proceedings of ACM-SIGKDD, pp. 847–855 (2013)
15. Feurer, M., Klein, A., Eggensperger, K., Springenberg, J., Blum, M., Hutter, F.: Efficient and robust automated machine learning. In: Advances in Neural Information Processing Systems, pp. 2962–2970 (2015)
16. Candelieri, A., Archetti, F.: Global optimization in machine learning: the design of a predictive analytics application. Soft. Comput. **23**, 2969–2977 (2018)
17. Elsken, T., Metzen, J.H., Hutter, F.: Neural Architecture Search: a Survey. J. Mach. Learn. Res. **20**(55), 1–21 (2019)
18. Galuzzi, B., Perego, R., Candelieri, A., Archetti, F.: Bayesian optimization for full waveform inversion. In: New Trends in Emerging Complex Real Life Problems, pp. 257–264 (2018)
19. Sergeyev, Y.D., Pugliese, P., Famularo, D.: Index information algorithm with local tuning for solving multidimensional global optimization problems with multiextremal constraints. Math. Program. **96**(3), 489–512 (2003)
20. Paulavičius, R., Žilinskas, J.: Advantages of simplicial partitioning for Lipschitz optimization problems with linear constraints. Optim. Lett. **10**(2), 237–246 (2016)
21. Strongin, R.G., Sergeyev, Y.D.: Global Optimization with Non-convex Constraints: Sequential and Parallel Algorithms, vol. 45, pp. 379-418. Springer, Berlin (2013)
22. Grishagin, V., Israfilov, R.: Multidimensional constrained global optimization in domains with computable boundaries. In: CEUR Workshop Proceedings. 1513: Proceedings of the 1st Ural Workshop on Parallel, Distributed, and Cloud Computing for Young Scientists (Ural-PDC 2015), Yekaterinburg (2015)
23. Di Pillo, G., Grippo, L.: Exact penalty functions in constrained optimization. SIAM J. Control Optim. **27**(6), 1333–1360 (1989)
24. Di Pillo, G., Lucidi, S., Rinaldi, F.: A derivative-free algorithm for constrained global optimization based on exact penalty functions. J. Optim. Theory Appl. **164**(3), 862–882 (2015)
25. Di Pillo, G., Liuzzi, G., Lucidi, S., Piccialli, V., Rinaldi, F.: A DIRECT-type approach for derivative-free constrained global optimization. Comput. Optim. Appl. **65**(2), 361–397 (2016)
26. Liu, J., Teo, K.L., Wang, X., Wu, C.: An exact penalty function-based differential search algorithm for constrained global optimization. Soft. Comput. **20**(4), 1305–1313 (2016)
27. Donskoi, V.I.: Partially defined optimization problems: an approach to a solution that is based on pattern recognition theory. J. Sov. Math. **65**(3), 1664–1668 (1993)
28. Rudenko, L.I.: Objective functional approximation in a partially defined optimization problem. J. Math. Sci. **72**(5), 3359–3363 (1994)
29. Sergeyev, Y.D., Kvasov, D.E., Khalaf, F.M.: A one-dimensional local tuning algorithm for solving GO problems with partially defined constraints. Optim. Lett. **1**(1), 85–99 (2007)
30. Bachoc, F., Helbert, C., Picheny, V.: Gaussian process optimization with failures: classification and convergence proof. HAL id: hal-02100819, version 1 (2019)
31. Sacher, M., Duvigneau, R., Le Maitre, O., Durand, M., Berrini, E., Hauville, F., Astolfi, J.A.: A classification approach to efficient global optimization in presence of non-computable domains. Struct. Multidiscip. Optim. **58**(4), 1537–1557 (2018)
32. Digabel, S.L., Wild, S.M.: A taxonomy of constraints in simulation-based optimization. arXiv preprint arXiv:1505.07881 (2015)
33. Hernández-Lobato, J.M., Gelbart, M.A., Adams, R.P., Hoffman, M.W., Ghahramani, Z.: A general framework for constrained Bayesian optimization using information-based search. J. Mach. Learn. Res. **17**(1), 5549–5601 (2016)
34. Gorji Daronkolaei, A., Hajian, A., Custis, T.: Constrained Bayesian optimization for problems with piecewise smooth constraints. In: Advances in Artificial Intelligence: 31st Canadian Conference on Artificial Intelligence, Canadian AI 2018, Toronto, ON, Canada, May 8–11, 2018, Proceedings, 31, pp. 218–223 (2018)
35. Picheny, V., Gramacy, R.B., Wild, S., Le Digabel, S.: Bayesian optimization under mixed constraints with a slack-variable augmented Lagrangian. In: Advances in Neural Information Processing Systems, pp. 1435–1443 (2016)
36. Feliot, P., Bect, J., Vazquez, E.: A Bayesian approach to constrained single-and multi-objective optimization. J. Glob. Optim. **67**(1–2), 97–133 (2017)

37. Gramacy, R.B., Lee, H.K.M., Holmes, C., Osborne, M.: Optimization under unknown constraints. Bayesian Stat. **9**, 229 (2012)
38. Bernardo, J., Bayarri, M.J., Berger, J.O., Dawid, A.P., Heckerman, D., Smith, A.F.M., West, M.: Optimization under unknown constraints. Bayesian Stat. **9**(9), 229 (2011)
39. Hernández-Lobato, J.M., Gelbart, M.A., Hoffman, M.W., Adams, R.P., Ghahramani, Z.: Predictive entropy search for Bayesian Optimization with unknown constraints. In: Proceedings of the 32nd International Conference on Machine Learning, 37 (2015)
40. Sui, Y., Gotovos, A., Burdick, J., Krause, A.: Safe exploration for optimization with Gaussian processes. In: International Conference on Machine Learning, 997–1005 (2015)
41. Sui, Y., Zhuang, V., Burdick, J.W., Yue, Y.: Stagewise Safe Bayesian Optimization with Gaussian Processes. arXiv preprint arXiv:1806.07555 (2018)
42. Scholkopf, B., Smola, A.J.: Learning with kernels: support vector machines, regularization, optimization, and beyond. MIT Press, Cambridge (2001)
43. Steinwart, I., Christmann, A.: Support vector machines. Springer, Berlin (2008)
44. Basudhar, A., Dribusch, C., Lacaze, S., Missoum, S.: Constrained efficient global optimization with support vector machines. Struct. Multidiscip. Optim. **46**(2), 201–221 (2012)
45. Tsai, Y.A., Perego, R., Pedrielli, G., Zabinsky, Z.B., Candelieri, A., Huang, H., Mathesen, L.: Stochastic Optimization for Feasibility Determination: An Application to Water Pump Operation in Water Distribution Network. In: Winter Simulation Conference 2018 (WSC 2018), Winter Simulation Conference 2018, December 9–12, Gothenburg, Sweden
46. Candelieri, A., Perego, R., Archetti, F.: Bayesian optimization of pump operations in water distribution systems. J. Glob. Optim. **71**(1), 213–235 (2018)
47. Letham, B., Karrer, B., Ottoni, G., Bakshy, E.: Constrained Bayesian optimization with noisy experiments. Bayesian Anal. **14**(2), 495–519 (2018)
48. Candelieri, A., Archetti, F.: Sequential model based optimization with black-box constraints: Feasibility determination via machine learning. In: AIP Conference Proceedings 2070(1), 020010, AIP Publishing (2019)
49. Sergeyev, Y.D., Kvasov, D.E., Mukhametzhanov, M.S.: Emmental-type GKLS-based multiextremal smooth test problems with non-linear constraints. In: R. Battiti et al. (Eds.): LION 2017, LNCS 10556, pp. 383–388, Springer, Cham (2017)
50. Rasmussen, C.E., Williams, C.K.: Gaussian processes for machine learning, vol. 38, pp. 715–719. The MIT Press, Cambridge (2006)
51. Wu, J., Poloczek, M., Wilson, A.G., Frazier, P.: Bayesian optimization with gradients. In: Advances in Neural Information Processing Systems, pp. 5267–5278 (2017)
52. Brochu, E., Cora, V.M., De Freitas, N.: A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. arXiv preprint arXiv:1012.2599 (2010)
53. Hartfiel, D.J., Curry, G.L.: On optimizing certain nonlinear convex functions which are partially defined by a simulation process. Math. Program. **13**(1), 88–93 (1977)
54. Srinivas, N., Krause, A., Kakade, S.M., Seeger, M.: Gaussian process optimization in the bandit setting: no regret and experimental design. In: Proceedings of International Conference on Machine Learning, pp. 1015–1022 (2010)
55. Neve, A.G., Kakandikar, G.M., Kulkarni, O.: Application of Grasshopper Optimization Algorithm for Constrained and Unconstrained Test Functions. Int. J. Swarm Intel. Evol. Comput. **6**(165), 2 (2017)
56. Simionescu, P.A., Beale, D.G.: New concepts in graphic visualization of objective functions. In: ASME 2002 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, pp. 891–897 (2002)
57. Mishra, S.K.: Some new test functions for global optimization and performance of repulsive particle swarm method. MPRA Paper No. 2718 (2008)
58. Picheny, V., Wagner, T., Ginsbourger, D.: A benchmark of kriging-based infill criteria for noisy optimization. Struct. Multidiscip. Optim. **48**(3), 607–626 (2012)
59. Huang, D., Allen, T.T., Notz, W.I., Zheng, N.: Global optimization of stochastic black-box systems via sequential Kriging meta-models. J. Glob. Optim. **3**(34), 441–466 (2006)
60. Hoffman, M.D., Brochu, E., De Freitas, N.: Portfolio Allocation for Bayesian Optimization, In: UAI, pp. 327–336 (2011)
61. Grishagin, V.A.: Operational characteristics of some global search algorithms. Probl. Stoch. Search **7**, 198–206 (1978)