Delaunay-Based Global Optimization in Nonconvex Domains Defined by Hidden Constraints

Shahrouz Ryan Alimo, Pooriya Beyhaghi and Thomas R. Bewley

Abstract This paper introduces a new surrogate-based optimization algorithm to optimize a deterministic objective function with non-computable constraint functions (a.k.a. hidden constraints). Both the objective function and the feasible domain are defined within a known rectangular domain. The objective function might be nonconvex, computationally expensive, and without analytic expression. Moreover, the feasible domain boundaries are not explicitly defined, but can be determined via oracle calls (feasible or not) and learned as the algorithm proceeds. To solve this class of optimization problems, the proposed algorithm, in each iteration, approximates the feasible domain boundary by incorporating a Support Vector Machine (SVM) classifier model as an approximation for the non-computable constraint function, which characterizes the feasible domain. The uncertainty associated with this surrogate is modeled using an artificially-generated uncertainty function built on the framework of Delaunay triangulation. This work extends the Delaunay-based optimization algorithm with nonconvex constraints, dubbed Δ-DOGS(Ω), and extends this approach to estimate the feasible domain with binary oracle calls. Similarly, this algorithm at each iteration determines a minimizer of the objective function surrogate model with the highest probability of being feasible. We evaluate the performance of the algorithm through the numerical experiments on a representative test problem.

S. R. Alimo (✉) · T. R. Bewley
UC San Diego, San Diego, CA, USA
e-mail: shahrouz.ryan.alimo@gmail.com

T. R. Bewley
e-mail: bewley@ucsd.edu

P. Beyhaghi
ASML-Cymer San Diego, San Diego, CA, USA
e-mail: p.beyhaghi@gmail.com

E. Andrés-Pérez et al. (eds.), *Evolutionary and Deterministic Methods for Design Optimization and Control With Applications to Industrial and Societal Problems*, Computational Methods in Applied Sciences 49, https://doi.org/10.1007/978-3-319-89890-2_17

Introduction

This paper aims at solving the optimization problems of the form

$$\begin{aligned} &\text{minimize} f(x) \quad \text{with} \quad x \in \Omega := L_c \cap L_s \subseteq \mathbb{R}^n \quad \text{where} \\ &L_c = \{x | c(x) \leq 0\}, \quad L_s = \{x | a \leq x \leq b\}, \end{aligned} \tag{1}$$

where $f(x) : \mathbb{R}^n \rightarrow \mathbb{R}$, and Ω is defined with hidden constraint functions. The only available measurements are of the sign$\{c(x)\}$. Then $c(x)$ is treated as binary measurement (e.g., ignoring information about the distance to the boundary feasible region); therefore measurements indicate "feasible" and "infeasible" regions Lee et al. (2011).

Motivation

Having a feasible domain that is not defined explicitly is common in some industrial applications such as shape optimization Gramacy et al. (2016) and chemical reactions Gelbart et al. (2018). In these situations, the optimums of the objective function must be further evaluated, and the solutions can be acceptable or unacceptable. That is, the feasible domain is only available through costly oracle calls. Nevertheless, in working conditions, these applications need to be tuned via a limited set of adjustable continuous parameters; for instance, in shape optimization, problems can be solved using only a handful of adjustable parameters usually modeled with $n < 10$. Most of the trial-and-error approaches (e.g) to do so could be both computationally expensive and time consuming. Thus, there is an ever-increasing need to develop efficient frameworks that could limit the number of adjustments needed and to automate the work of these complex systems.

The objective of this work is to develop a new optimization method which is designed for when the objective function is expensive to calculate. In addition, the constraint violation comes from the same process as the objective function, but is accessible through binary oracle calls (acceptable or unacceptable). In these settings, the response of the simulator/system is whether a constraint is violated or not, and no further information about the simulator is given Lee et al. (2011). Thus, the proposed method needs to simultaneously estimate the feasible region (constraint region) and solve the minimization problem.

Many modern optimization approaches for shape optimization of computer-aided designs converge without derivative information and require only weak regularity conditions Gramacy et al. (2016), Alimo et al. (2017), Marsden et al. (2004), Moghadam et al. (2012). When the constraint function is hidden, a simulation displays a flag indicating whether a capacitor has been become full during the simulation, but we know neither when this occurred nor the level of capacitor charge. Such

problems are some of the most challenging optimization problems, and most of the existing approaches rely on heuristic approaches Digabel and Wild (2015).

One of the first schemes for solving such a family of problems was introduced by Conn et al. (1998), where they considered the trust-region subproblem. The authors also described virtual constraints as non-computable constraints and recommend using an extreme-barrier approach. This approach is based on restricting the solution to inside the trust-region subproblem, and based on whether the point is feasible or not, the trust region is adjusted. Further improvement is made by using an extreme-barrier approach.

Regarding other approaches, Gramacy et al. (2016) developed a statistical approach based on Gaussian processes and Bayesian learning to both approximate the unknown function and estimate the probability of meeting the constraints. In another approach, Lee et al. (2011) forced the problem (1) into an existing statistical framework by using treed Gaussian processes for response surface prediction, and random forests for constraint violation prediction. Finally, there are some recently introduced algorithms Picheny et al. (2016), Gelbart et al. (2018) that are based on Augmented Lagrange, and they try to improve the performance of such schemes. However, most of these algorithms are statistical-based methods, and they usually consider the underlying signal as a realization from a random process. Moreover, most of these algorithms are not globally convergent, and they only have the potential to search globally.

A Delaunay-based optimization algorithm, Δ-DOGS, was a recently developed derivative-free optimization algorithm. This algorithm was extended in Δ-DOGS(Ω) in order to solve, with remarkable efficiency, optimization problems with nonconvex and computationally expensive objective and constraint functions Alimo et al. (2017), Alimo et al. (2018). This new algorithm is provably convergent under the appropriate assumptions. This algorithm can be classified as a response surface method which iteratively solves a subproblem based on interpolations not only of the objective and constraint functions over existing datapoints, but also a synthetic model of the uncertainty of these interpolants, which itself is built on the framework of a Delaunay triangulation over existing datapoints. Unlike other response surface methods, this algorithm can employ any well-behaved interpolation strategy.

This paper introduces a new scheme based on Δ-DOGS(Ω) that solves problems where the feasible domain is not known and can only be approximated using a number of oracle calls within the parameter space. That is, this work solves problems where the constraint functions are hidden functions as (2).

The remainder of the paper is organized as follows. Section 2 briefly reviews the optimization algorithm Δ-DOGS(Ω) that was proposed for solving problems with nonconvex and computationally expensive constraint functions. Section 3 describes the extension of this method to solve problems with hidden constraints. Section 4 illustrates the behaviour of the new method on a representative test problem. Some conclusions are made in Sect. 5.

Review of Δ-DOGS(Ω)

Delaunay-based optimization is a generalizable family of practical, efficient, and provably-convergent derivative-free algorithms designed for a range of nonconvex optimization problems with expensive function evaluations Beyhaghi et al. (2015), Beyhaghi and Bewley (2016). This framework, dubbed Δ-DOGS, was extended by an algorithm Δ-DOGS(Ω) in order to solve optimization problems with both nonconvex and computationally expensive objective and constraint functions Alimo et al. (2017), Alimo et al. (2018). Δ-DOGS(Ω) solves problems in the form of:

$$\begin{aligned} &\text{minimize} f(x) \quad \text{with} \quad x \in \Omega := L_c \cap L_s \subseteq \mathbb{R}^n \quad \text{where} \\ &L_c = \{x | c_\ell(x) \leq 0\}, \quad L_s = \{x | a \leq x \leq b\}, \end{aligned} \tag{2}$$

where both $f(x)$ and $c_\ell(x)$ for $\ell = 1, \ldots, m$ are twice differentiable and possibly nonconvex functions which map $\mathbb{R}^n \to \mathbb{R}$ within the search domain L_s. The optimization problem (2) has two sets of constraints:

(a) a set of $2n$ bound constraints that characterize the n-dimensional box domain $L_s = \{x | a \leq x \leq b\}$, dubbed the *search domain*, and
(b) a set of m possibly nonlinear inequality constraints $c_\ell(x) \leq 0$ that together characterize the possibly nonconvex domain L_c, dubbed the *constraint domain*.

The *feasible domain* is the intersection of these two domains, $\Omega := L_s \cap L_c$.

In many application based problems, we are seeking a feasible point $x \in \Omega$ such that $f(x) \leq f_0$, where f_0 is a target value. In this work, we assume that there is a known target value f_0, which is achievable; i.e., $\exists x \in L_s$ such that $f(x) \leq f_0$ and $c_\ell(x) \leq 0$ for all $\ell = 1, \ldots, m$. The $c_\ell(x)$ are nonlinear, computationally expensive constraint functions. However, it is worth noting that the present algorithm can be easily extended to problems for which a target value and constraint violation thresholds are not available, as in Algorithm 1 of Alimo et al. (2018).

We assume that $f(x)$ and $c_\ell(x)$ are computable everywhere in L_s computationally expensive, and possibly nonconvex. For the purpose of the scheme development, we assume that $f(x)$ and $c_\ell(x)$e twice differentiable. Also, the gradient information for $f(x)$, or its estimate, is usually not available. Moreover, we consider the optimization problems with few adjustable parameters $n \leq 10$. Before presenting the algorithm, we introduce some preliminary concepts:

Definition 1 Given S^k as a set of points that includes the vertices of domain L_s at iteration k of Δ-DOGS(Ω), we define $p^k(x)$ and $g_1^k(x), \ldots, g_m^k(x)$ as a set of successive interpolations for the objective and constraint functions $f(x) and c_1(x) \ldots, c_m(x)$, respectively, at iteration k. Consider

$$T^k(x) = \max \left[p^k(x) - f_0,\ g_1^k(x), \ldots,\ g_m^k(x) \right], \tag{3}$$

then the **continuous search function** is defined as

$$s_c^k(x) = \begin{cases} T^k(x)/e^k(x), & \text{if } T^k(x) \geq 0, \\ T^k(x), & \text{otherwise.} \end{cases} \tag{4}$$

Δ-DOGS(Ω) estimates (2) with a set of surrogates, and in the situation where there exists a target value for the objective function such that $f_0 \leq f(x)$ and there are m different nonconvex constraint functions defining the feasible domain, then the algorithm iteratively evaluates the minimizer of (4). This method is an iterative algorithm, and at each iteration, it generates a set of points to estimate the objective functions and the feasible domain. Using the set of available datapoints, the objective functions are approximated with interpolation/regression models, and the uncertainty of the interpolants is quantified using an artificially generated uncertainty model based on the Delaunay triangulation framework. The most expensive part in this calculation is the function evaluation process.

Algorithm 1 Δ-DOGS (Ω) for accurate objective and constraint function evaluations Alimo et al. (2018).

1. Set $k = 0$. Take the set of initialization points S_0 as all vertices of the feasible domain Ω
2. Calculate (or, for $k > 0$, update) an appropriate interpolating functions $p(x)$, $g_\ell(x)$ through all points in S_k
3. Calculate (or, for $k > 0$, update) a Delaunay triangulation Δ^k over all of the points in S_k
4. Find $x_k \in \Omega$ as a global minimizer of $s(x)$ subject to $CS_\ell(x) \leq 0$, for $\ell = 1, 2, \cdots, m$.
5. Calculate $f(x)$, $c_\ell(x)$ at x_k, and take $S_{k+1} = S_k \cup x_k$.

In convergence analyses for Δ-DOGS(Ω), the objective and constraint functions were assumed to be twice differentiable with the search domain L_s.

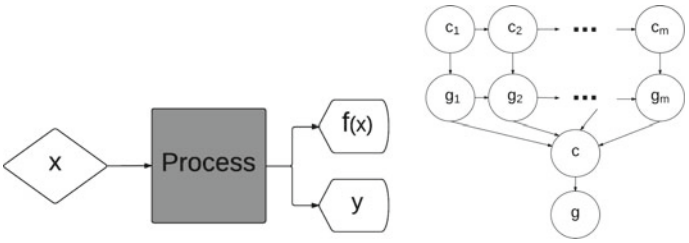
Furthermore, we can observe for Δ-DOGS(Ω), Algorithm 1, that

- if $f_0 \geq f(x^*)$, Δ-DOGS (Ω) generates an infinite sequence of points whose limit points are characterized by objective function values less or equal to f_0, or
- if $f_0 < f(x^*)$, Δ-DOGS (Ω) generates an infinite sequence of points which is dense everywhere in the feasible domain.

Remark 1 With existence of a target value f_0 for the objective function, Algorithm can find a point $x \in \Omega$ which is feasible and $f(x) \leq f_0$, if such point exists. In fact, any limit point of S is feasible, and its objective function is less than f_0. If there is no point in which $f(x) \leq f_0$ and x is feasible, Algorithm will go dense everywhere in the feasible domain Ω.

In the following section we extend Δ-DOGS (Ω) for the cases where the evaluation of the constraint function $c(x)$ is only limited to its sign as shown in Fig. 1.

Unlike the previous work Alimo et al. (2018) for Δ-DOGS(Ω) where the function evaluation was computable, in the next section, the constraint violation is considered



(a) Solver outputs. (b) Hidden constraints.

Fig. 1 Function evaluation process for the problems in form of (1). **a** Each constraint violation is measured only with feasible or infeasible as $y_i = \text{sign}\{c(x^{(i)})\}$. **b** The feasible domain is modeled using a surrogate model g

non-computable and is only limited to an oracle call or a binary measurement (feasible or unfeasible). In this problem, we are not able to measure the constraint function, and we are limited to finding the sign of c_ℓ for $\ell = 1, \ldots, m$, and the point $x^{(i)}$ is either feasible or unfeasible.

Quantifying Constraint Violation Using SVM

In this section, we extend the Δ-DOGS(Ω) algorithm to solve the optimization problem with the feasible domain that is characterized with binary oracle calls (feasible or not). For each set of parameters the constraint violation is determined by

$$y(x_i) = \begin{cases} -1, & \text{if } x_i \text{ is feasible or } \quad c(x) \leq 0 \\ +1, & \text{if } x_i \text{ is infeasible or } \quad c(x) > 0. \end{cases} \tag{5}$$

We assume that the information about the feasible domain boundaries are only available through function (5). In other words, since the constrains are not defined explicitly, we only can determine if a point of interest is within the feasible domain or not. This situation is similar to the binary classification problems where the training points (data) are divided into two classes. The final goal of a classification problem is to predict the class of a new candidate point. In this work, we borrow that idea and extend it to use in the optimization problem (2) to estimate the boundary of feasibility as the algorithm proceeds.

There are a wide variety of classifiers in the machine learning literature such as Perceptron, Artificial Neural Network (ANN), and Support Vector Machine (SVM).

Perceptron is a linear classifier that tries to find a hyperplane that separates the labeled training data points into two classes so that points with the same label stays in one side of the hyperplane. The Perceptron algorithm starts with an initial hyperplane. At each iteration of the algorithm it processes a point from the training set and

update the weight of parameters to characterize the optimal hyperplane calssifier. The algorithm is suitable for online learning problems where the training data points are given sequentially. In addition, computationally it is very appealing. However, it requires a relatively large training set and the final hyperplane is not necessarily the best hyperplane that separates the two classes Cortes and Vapnik (1995). Moreover, since this problem is not linearly separable using Perceptron is not practical.

ANN Goodfellow et al. (2016) impose multiple-layer linear classifiers that can classify data points using only a handful of features, but they include many undetermined and hidden layers. The drawback for this approach is that there are many tuning parameters to characterize these hidden layers correctly. These tuning parameters are problem specific, and they require a large amount of data to train the hidden layers of the network Bengio et al. (2015).

SVM Cortes and Vapnik (1995), Cherkassky and Ma (2004) is a linear classifier, which aims at determining the maximum margin hyperplane. SVM classifier transfers features into a higher dimensional space using appropriate kernels and then fits a linear classifier. As a result it can also separate the data points that are not linearly separable. Overall, the SVM classifier is computed by solving a quadratic programming optimization problem.

At each iteration, the Δ-DOGS(Ω_H) search for a minimizer of the optimization problem which has the highest probability of being inside the feasible domain. The oracle calls are assumed to be computationally expensive therefor the adapted classifier should be able to estimate the boundaries using fewer samples. In this paper, we propose a SVM-based approach to approximate the feasible domain of solutions at each iteration of the optimization algorithm.

We assume that the boundaries of the feasible domain are twice differentiable with a bounded Lipschitz norm. In other words, there exists an unknown underlying function in which $\text{sign}\{c(x)\} = \text{sign}\{g(x)\}$ for all $x \in L_s$. To be able to approximate a wide variety of boundary functions, we consider the Radial Basis Functions (RBF) as the kernel of the SVM. That is

$$g(x) = \sum_{i=1}^{d} w_i \, \phi_i(x) + b, \tag{6}$$

where we consider b as a bias term and $\phi(x)$ as radial basis kernel functions that denote the feature space transformation. Let $y_i = \text{sign}\{g(x_i)\}$, then the distance of a point x_i to the decision surface $g(x_i)$ is given by

$$\frac{y_i \, g(x_i)}{\|w\|} = \frac{y_i \, (w^T \, \phi(x) + b)}{\|w\|} > 0.$$

Note that x_i for $i = 1, \ldots, N$ are evaluated points in the feature (parametric) space at the Nth iteration. In this way, every point x_i can be transformed into a d-D space such that $X_i = [\phi_1(x_i), \phi_2(x_i), \ldots, \phi_d(x_i)]$.

The optimal hyperplane is determined by solving a quadratic programming (QP) as follows:

$$\min_{z\in\mathbb{R}^{d+1}} z^T L z, \quad \text{subject to} \quad Y.(Az) \geq 1, \quad \text{where}$$
$$L = \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix}, \qquad A = \begin{bmatrix} F & 1^T \\ 1 & 0 \end{bmatrix},$$
$$Y = \begin{bmatrix} \text{sign}\{c(x_1)\} \\ \vdots \\ \text{sign}\{c(x_N)\} \\ 0 \end{bmatrix}, \qquad z = \begin{bmatrix} w_1 \\ \vdots \\ w_d \\ b \end{bmatrix}.$$

where $F_{i,j} = \phi_i(x_j) = \varphi(|x_i - x_j|)$ and $1 = ([1, \ldots, 1]^T)_{N\times 1}$. Performance depends on the choice of basis functions $\phi(x)$ used to leverage the evaluated dataset, and the choice of SVM kernels are problem-dependent. We set

$$\phi_i(x) = \varphi(r) \quad \text{, where} \quad r = |x_i - x|$$

The most well-known RBF models are the Gaussian kernel model $\varphi(r) = e^{-r^2/\sigma^2}$, the polynomial model $\varphi(r) = r^3$, and the inverse multi-quadratic kernel model $\varphi(r) = 1/\sqrt{\sigma^2 + r^2}$.

The main challenge in optimization with virtual constraints is that there are many models of g that could successfully classify the two classes from each other, and these approximated constraint functions could have a very different range of values from one another. The variety in appropriate g models stems from the fact that we only have access to the sign of $c(x)$. As a result, the inclusion of more data points from specific regions can sometimes lead to estimated g models deviating from the true hidden function. To control this deviation, we scale the estimated constraint function using the initial training data points in order to have the same range of variations as the objective function $f(x)$.

Δ-DOGS(Ω_H)

The new algorithm is designed based on the Δ-DOGS(Ω). Since the constraints are hidden, an approximation (6) based on the SVM is considered for the model of constraints. The search function is defined as

$$s_c(x) = \max\{\frac{p(x) - f_0}{r_f\, e(x)}, \ \frac{g(x)}{r_g\, e(x)}\} \tag{7}$$

where $g(x)$ is modeled as (6) and r_f and r_g are constants to make the constraint search function model in the same range of variation as the objective search function.

Δ-DOGS(Ω_H) algorithm depends upon a handful of adjustable parameters, the selection of which affects its rate of convergence. The remainder of this section discusses heuristic strategies to tune these algorithm parameters, noting that this tuning is an application-specific problem, and alternative strategies (based on experiment or intuition) might lead to more rapid convergence for certain problems.

The first task encountered during the setup of the optimization problem is the definition of the design parameters and search domain L_s. Note that the feasible domain considered during the optimization process is characterized by simple upper and lower bounds for each design parameter; normalizing all design parameters to lie between 0 and 1 is often beneficial Alimo et al. (2018), Beyhaghi and Bewley (2018).

The second challenge is to scale the objective function $f(x)$ and the hidden constraint function $g(x)$ themselves, such that the range of the normalized functions $f(x)$ and $g(x)$ over the search domain L_s are the same and about unity.

If an estimate of the actual range of $c(x)$ (the same as $g(x)$) is not available a prior, we may estimate it at any given iteration using the available measurements.

Results

The test function is considered as a quadratic objective function, given by the distance from a point in the search domain L_s, and is defined over an n-dimensional space, subject to a nonlinear inequality constraint. The feasible domain is generated using the sign of a Rastrigin function, defining a disconnected feasible domain characterized by 2^n distinct "islands" within the search domain:

$$\min_{x \in L_s} \quad f(x) = \|x - x_0\|^2 - 0.024\, n, \tag{8a}$$

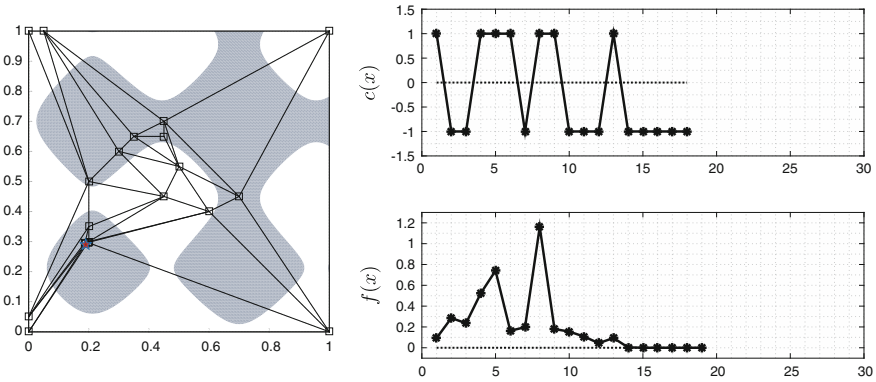
$$\text{subject to} \quad \text{sign}\{c(x)\} < 0, \tag{8b}$$

$$c(x) = \frac{n}{12} + \frac{1}{10} \sum_{i=1}^{n} \left\{ 4\,(x_i - 0.7)^2 - 2\,\cos\Big(4\pi\,(x_i - 0.7)\Big) \right\},$$

$$0 \le x_1, x_2, \ldots, x_n \le 1. \tag{8c}$$

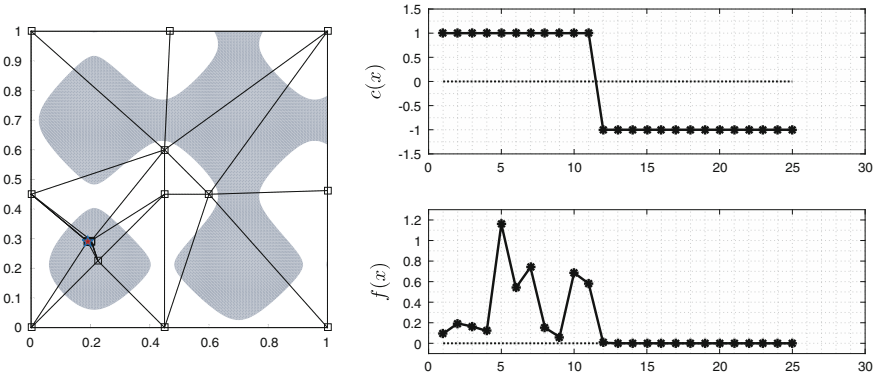
This problem has 2^n local minima, including the unique global minimum where $x_0 = [0.19, 0.29]^T$ for $n = 2$ with $f(x^*) = f(x_0) = 0$.

The results show that Δ-DOGS(Ω_H), the newly introduced method, has the potential to identify the global minimizer of the objective function under the feasible region. It is observed that the choice of kernel function acts as a crucial factor in enabling the global convergence of the optimization method. Figures 2 and 3 show the results using a piece-wise linear kernel. This is an appropriate choice since the underlying model of constraint function range will not change exponentially when the algorithm detects a feasible region, as was the case when a cubic kernel was used. In addition, use of a cubic kernel did not lead to global convergence.



(a) Position of points (b) Convergence history

Fig. 2 Nonconvex problem for $n = 2$ with vertices as initial points. The gray region is the feasible domain. The objective function is the distance function from the global minimizer showed by red star $x^* = [0.19, 0.29]^T$



(a) Position of points (b) Convergence history

Fig. 3 Nonconvex problem for $n = 2$ with $\{x_{0,i} = 0.35, x_{0,i} + 0.1\, e_i\}$ where e_i is the ith main coordinate direction. The gray region is the feasible domain and the objective function is the distance function from $x^* = [0.19, 0.29]^T$

Conclusions

In this work, we presented a new derivative-free algorithm for the optimization of expensive cost functions subject to box constraints and hidden type of constraints, in which a design point can be labeled as feasible or not feasible. We modeled the hidden constraints with the popular classification techniques that is from the machine learning literature. The well known techniques of SVMs are applied in a global optimization framework.

As a future work, we compare the results from Artificial Neural Network and leverage deep learning techniques to quantify $g(x)$. Furthermore, the presented optimization method will be applied to an application-based problem, and we will study the behaviour of $g(x)$ in different problems.

References

Alimo SR, Beyhaghi P, Bewley TR Delaunay-based derivative-free optimization via global surrogates, part iii: nonconvex constraints. J Glob Optim

Alimo SR, Cavaglieri D, Beyhaghi P, Bewley TR (2017) Design of imexrk time integration schemes via delaunay-based derivative-free optimization. Journal of Global Optimization, under preparation

Alimo S, Beyhaghi P, Meneghello G, Bewley T (2017) Delaunay-based optimization in cfd leveraging multivariate adaptive polyharmonic splines (maps). In: 58th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, 0129

Bengio Y, Goodfellow IJ, Courville A (2015) Deep learning. Nature 521:436–444

Beyhaghi P, Bewley T (2016) Delaunay-based derivative-free optimization via global surrogates, part ii: convex constraints. J Glob Optim, under review

Beyhaghi P, Bewley T Delaunay-based via lattice-based optimization algorithm. J Glob Optim

Beyhaghi P, Cavaglieri D, Bewley T (2015) Delaunay-based derivative-free optimization via global surrogates, part i: linear constraints. J Glob Optim 63:1–52

Cherkassky V, Ma Y (2004) Practical selection of svm parameters and noise estimation for svm regression. Neural networks 17(1):113–126

Conn A, Scheinberg K, Toint P (1998) A derivative free optimization algorithm in practice. In: 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, 4718

Cortes C, Vapnik V (1995) Machine learning. Support-vector networks 20(3):273–297

Digabel SL, Wild SM (2015) A taxonomy of constraints in simulation-based optimization. arXiv preprint arXiv:1505.07881

Gelbart MA, Snoek J, Adams RP (2014) Bayesian optimization with unknown constraints

Goodfellow I, Bengio Y, Courville A (2016) Deep learning. MIT press, Cambridge

Gramacy RB, Gray GA, Le Digabel S, Lee HK, Ranjan P, Wells G, Wild SM (2016) Modeling an augmented lagrangian for blackbox constrained optimization. Technometrics 58(1):1–11

Lee H, Gramacy R, Linkletter C, Gray G (2011) Optimization subject to hidden constraints via statistical emulation. Pac J Optim 7(3):467–478

Marsden AL, Wang M, Dennis JE Jr, Moin P (2004) Optimal aeroacoustic shape design using the surrogate management framework. Optim Eng 5(2):235–262

Moghadam ME, Migliavacca F, Vignon-Clementel IE, Hsia T-Y, Marsden AL (2012) Optimization of shunt placement for the norwood surgery using multi-domain modeling. J Biomech Eng 134(5):051002

Picheny V, Gramacy RB, Wild S, Le Digabel, S (2016) Bayesian optimization under mixed constraints with a slack-variable augmented lagrangian. In: Advances in neural information processing systems, pp 1435–1443