Journal of Global Optimization (2020) 77:743–776
https://doi.org/10.1007/s10898-019-00854-2

Delaunay-based derivative-free optimization via global surrogates. Part III: nonconvex constraints

Ryan Alimo[1,2] · **Pooriya Beyhaghi**[1] · **Thomas R. Bewley**[1]

Received: 8 March 2018 / Accepted: 30 October 2019 / Published online: 23 January 2020

Abstract
This paper introduces a Delaunay-based derivative-free optimization algorithm, dubbed Δ-DOGS(Ω), for problems with both (a) a nonconvex, computationally expensive objective function $f(x)$, and (b) nonlinear, computationally expensive constraint functions $c_\ell(x)$ which, taken together, define a nonconvex, possibly even disconnected feasible domain Ω, which is assumed to lie within a known rectangular search domain Ω_s, everywhere within which the $f(x)$ and $c_\ell(x)$ may be evaluated. Approximations of both the objective function $f(x)$ as well as the feasible domain Ω are developed and refined as the iterations proceed. The approach is practically limited to the problems with less than about ten adjustable parameters. The work is an extension of our original Delaunay-based optimization algorithm (see JOGO DOI: 10.1007/s10898-015-0384-2), and inherits many of the constructions and strengths of that algorithm, including: (1) a surrogate function $p(x)$ interpolating all existing function evaluations and summarizing their trends, (2) a synthetic, piecewise-quadratic uncertainty function $e(x)$ built on the framework of a Delaunay triangulation amongst existing datapoints, (3) a tunable balance between global exploration (large K) and local refinement (small K), (4) provable global convergence for a sufficiently large K, under the assumption that the objective and constraint functions are twice differentiable with bounded Hessians, (5) an Adaptive-K variant of the algorithm that efficiently tunes K automatically based on a target value of the objective function, and (6) remarkably fast global convergence on a variety of benchmark problems.

Keywords Response surface methods · Nonconvex constraints

✉ Ryan Alimo
shahrouz.ryan.alimo@gmail.com

Pooriya Beyhaghi
p.beyhaghi@gmail.com

Thomas R. Bewley
bewley@eng.ucsd.edu

[1] Flow Control and Coordinated Robotics Labs, UC San Diego, San Diego, USA

[2] Jet Propulsion Laboratory, California Institute of Technology, Pasadena, USA

1 Introduction

The problem considered in this paper is as follows:

$$\begin{aligned} &\text{minimize } f(x) \quad \text{with} \quad x \in \Omega := \Omega_c \cap \Omega_s \subseteq \mathbb{R}^n \quad \text{where} \\ &\Omega_c = \{x | c_\ell(x) \leq 0\,, \text{ for } \ell = 1, \ldots, m\}, \quad \Omega_s = \{x | a \leq x \leq b\}, \end{aligned} \tag{1}$$

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 Ω_s. The optimization problem (1) has two sets of constraints:

a. a set of $2n$ bound constraints that characterize the n-dimensional box domain $\Omega_s = \{x | a \leq x \leq b\}$, 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 Ω_c, dubbed the *constraint domain*.

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

Potential applications of an efficient optimization algorithm of this type include: (a) minimizing the ratio of lift to drag, while holding the lift coefficient constant, in the design optimization of airfoils [38] and hydrofoils [3]; (b) minimizing entropy generation, with a constant wall temperature or wall heat flux, in the optimization of finned-tube heat exchangers [48]; (c) optimization of cardiovascular stints [35,42]; (d) maximizing solar power plant efficiency [4,36]; and (e) parameter optimization in deep learning networks [54]. In such problems, the number of adjustable parameters of interest is generally quite low (less than 10). The objective and constraint functions for such problems often come from black-box software, such as computational fluid dynamics codes, that often take several hours or more to evaluate for any given set of parameters. The Hessian of the objective function in such problems is often, apparently, bounded (though this smoothness is difficult or impossible to establish mathematically), Further, in many application-based problems, there is an initial reference design that one seeks to improve by a certain amount. The present optimization algorithm is well suited for such problems.

Constrained optimization problems of the form given in (1), dubbed Nonlinear Inequality Problems (NIPs), have been studied widely using both derivative-based and derivative-free optimization strategies.

In the derivative-based setting, there are two main classes of approaches for NIPs. The first, Sequential Quadratic Programming (SQP) methods (see, e.g., [20,21]), impose a local quadratic model for the objective function and a local linear model for the constraint functions to estimate the location of the local minimum at each step. These models are defined based on the local gradient and Hessian of the objective function, and the local Jacobian of the constraint functions; thus, these methods are applicable only when these derivatives, or accurate approximations thereof, are available. SQP methods only converge to a KKT point, or to a local minimum, and special care is needed to guarantee such convergence [20,22]. Application of SQP methods to problems with nonconvex feasible domains leads to some especially difficult technical challenges. Effective implementations of Sequential Quadratic Programming (SQP) methods include SNOPT [20] and SQP Filter [17].

The other main class of derivative-based approaches for NIPs is penalty methods (see, e.g., [19]), which modify the objective function by adding a penalty term which is successively refined as the optimization proceeds. In this way, a series of *unconstrained* problems is considered, and these problems are iteratively adjusted as convergence is approached such that the iterative process eventually converges to a local minimizer of the original *constrained*

problem. There are two main types of penalty functions used in such approaches: quadratic penalty functions (see, e.g., [10,27,28]) and barrier functions (see, e.g., [18]).

Methods based on quadratic penalty functions add a smooth penalty *outside* the feasibility boundary. This penalty goes to zero at the feasibility boundary, and is made successively steeper near the feasibility boundary (that is, it increases without bound on the exterior of the feasible domain) as convergence is approached. Methods based on quadratic penalty functions do not require the identification of an initial feasible point during initialization, though the objective function is evaluated over a search domain that extends beyond the feasible domain, so these infeasible objective function computations need to be well-behaved.

Methods based on barrier functions add a smooth penalty *inside* the feasibility boundary. This penalty goes to infinity at the feasibility boundary, and is made successively steeper near the feasibility boundary (that is, it is diminished towards zero on the interior of the feasible domain) as convergence is approached. Methods based on barrier functions require the identification of an initial feasible point during initialization, though all subsequent function evaluations are feasible, so the objective function need not be well-behaved outside the feasible domain.

Derivative-based methods, though scaling to higher-dimensional optimization problems far better than derivative-free methods, have certain distinct challenges. They are usually not designed to find the global minimum of a nonconvex objective function. During the last two decades new spatial branch-and-bound methods, e.g., BARON [55], COUENNE [9], SCIP [2], ANTIGONE [41], that are able to find the global minimum of a nonconvex objective function were developed. However, these methods need an expression (or, a numerical approximation) of the gradients of the objective and constraint functions; such derivative information is often difficult or impossible to obtain.

Many derivative-free strategies for constrained optimization have been proposed. Some of these methods, like the downhill simplex and direct search methods [including the generalized pattern search (GPS) and mesh-adaptive direct search (MADS) algorithms], only assure convergence to a local minimum. Others, like simulated annealing, genetic algorithms, exhaustive search strategies, and response surface methods, provide convergence to the global minimum; however, most such methods perform global exploration steps in either a random or an exhaustive fashion, and are thus quite inefficient with function evaluations. Today, response surface methods are the most efficient class of optimization schemes in derivative-free settings that provide global convergence [26].

Direct search methods do not use any model for the objective or constraint functions, and the solution of the optimization problem is found based only on a series of exploratory function evaluations inside the feasible domain. GPS methods [56], which are well-known algorithms in this class, restrict all function evaluations to lie on an underlying grid that is successively refined. GPS methods were initially designed for unconstrained problems, but have been modified to address bound constrained problems [32], linearly-constrained problems [33], and smooth nonlinearly constrained problems [34]. MADS methods [1,5–8] are modified GPS methods which can handle nonsmooth constraints.

Response Surface Methods (RSMs) [8,14,16,23,52,60], on the other hand, leverage an underlying inexpensive model, or "surrogate", of the objective function.[1] Kriging interpolation [29,50,51] was initially used to develop this surrogate. This convenient choice provides both an interpolant and a model of the uncertainty of this interpolant. However, such correlation-based interpolation strategies have a number of numerical shortcomings

[1] This approach generalizes the SQP method, where a quadratic function is used to locally model the objective function, and linear function is used to locally model the constraints.

(see, e.g., the appendix of [12]). Thus, our group has explored [11,12] a new class of RSMs that can employ *any* interpolation strategy for the surrogate.

The first method in this class [12] was limited to linearly constraint domain since a mesh with simplices was required to construct the uncertainty function at each feasible point. Later [11] extends it to problems with nonlinear convex constraints. In this paper, we will extend the original algorithm to problems with nonconvex constraint. There are two main differences in this approach compare to the previous works: a) The function evaluation is not limited to the feasible points and a larger search space is considered. b) Unlike original algorithms which defines surrogate functions only for objective function, the new method defines surrogate function to estimate constraints as well.

The structure of this paper is as follows. Section 2 describes the main elements of the optimization algorithm itself. Section 3 analyzes its convergence properties, and describes the technical conditions needed to guarantee its convergence to a global minimizer. Section 4 presents a procedure to adjust the tuning parameter of the algorithm developed in Sect. 2 to maximize the speed of convergence if an estimate of the global minimum (but, not the global minimizer) is available. Section 5 discusses briefly the behavior of the algorithm proposed when the feasible domain is empty. In Sect. 6, the algorithm is applied to some benchmark optimization problems to illustrate its behavior. Conclusions are presented in Sect. 7.

2 Optimization algorithm

In this section, we present an algorithm to solve the optimization problem defined in (1), in a manner which efficiently handles inequality constraint functions $c_\ell(x)$ that are nonlinear and computationally expensive, and which (together with Ω_s) define a feasible domain Ω that may be nonconvex and, even, not connected.

2.1 Preliminary definitions

We first present some preliminary notions.

Definition 1 Consider the $(n+1)$ vertices $V_0,\ V_1, \ldots,\ V_n \in \mathbb{R}^n$ such that the vectors $(V_0 - V_1), (V_0 - V_2), \ldots, (V_0 - V_n)$ are linearly independent. The convex hull of these vertices is called a **simplex** (see, e.g., [15, p. 32]). Associated with this simplex, the **circumcenter** z is the point that is equidistant from all $n+1$ vertices, the **circumradius** r is the distance between z and any of the vertices V_i, and the **circumsphere** is the set of all points within a distance r from z.

Definition 2 If S is a set of points in $\mathbb{R}^n$, a **triangulation** of S is a set of simplices whose vertices are elements of S such that the following conditions hold:

- Every point in S is a vertex of at least one simplex in the triangulation. The union of all of these simplices fully covers the convex hull of S.
- The intersection of two different simplices in the triangulation is either empty or a k-simplex such that $k = 0, 1, \ldots, n-1$. For example, in the case of $n = 3$ dimensions, the intersection of two simplices (in this case, tetrahedra) must be an empty set, a vertex, an edge, or a triangle.

Definition 3 A **Delaunay triangulation** is a triangulation (see Definition 2) such that the intersection of the open circumsphere around each simplex with S is empty. This special class of triangulation, as compared with other triangulations, has the following properties:

– The maximum circumradius among the simplices is minimized.
– The sum of the squares of the edge lengths weighted by the sum of the volumes of the elements sharing these edges is minimized.

Definition 4 Let S be a set of points in Ω_s, including its vertices. Define Δ as a Delaunay triangulation (see [13]) of S. Then, for each simplex $\Delta_i \in \Delta$, the *local uncertainty function* $e_i(x)$ is defined as

$$e_i(x) = (r_i)^2 - \|x - z_i\|^2, \tag{2}$$

where z_i and r_i are the circumcenter and circumradius of the simplex Δ_i, respectively. The *global uncertainty function* $e(x)$ is then defined as

$$e(x) = e_i(x), \quad \forall x \in \Delta_i. \tag{3}$$

The functions $e_i(x)$ and $e(x)$ have a number of properties which are shown in Section 3 of [12], the most important of which, for the present purposes, are as follows.

Property 1 *The global uncertainty function $e(x)$ is piecewise quadratic, continuous, and Lipschitz, with Lipschitz constant $2\,R_{\max}$, where $R_{\max}$ is the maximum circumradius of the corresponding Delaunay triangulation.*

Property 2 *The Hessian of local uncertainty function $e_i(x)$ is equal to $-2\,I$.*

Property 3 *The global uncertainty function $e(x)$ is equal to the maximum of the local uncertainty functions $e_i(x)$, that is,*

$$e(x) = \max_{i=1,\ldots,E} e_i(x), \quad \forall x \in \Delta, \tag{4}$$

where E is the number of simplices in the triangulations.

Definition 5 Let S be a set of (both feasible and infeasible) points in Ω_s, including its vertices, at which the objective function $f(x)$ has been evaluated. Consider $p(x)$ as an interpolating function in Δ that interpolates the objective function $f(x)$ at all points in S. Then, for each simplex $\Delta_i \in \Delta$, the *local search function* $s_i(x)$ is defined as

$$s_i(x) = p(x) - K\,e_i(x), \tag{5}$$

where K is a tuning parameter. The *global search function* $s(x)$ is defined as

$$s(x) = s_i(x) \qquad \forall x \in \Delta_i, \tag{6}$$

where Δ_i is the i'th simplex in Δ. Note that

$$s(x) = \min_{i=1,\ldots,E} s_i(x) \qquad \forall x \in \Delta.$$

Definition 6 Consider $g_1(x), g_2(x), \ldots, g_m(x)$ as interpolating functions in Δ that interpolate the constraint functions $c_1(x), c_2(x), \ldots, c_m(x)$, respectively, through the points in S. Then, for each simplex $\Delta_i \in \Delta$, the *ℓ'th local constraint search function* $\tilde{s}_{\ell,i}(x)$ is defined as

$$\tilde{s}_{\ell,i}(x) = g_\ell(x) - K\,e_i(x) \le 0, \qquad \ell = 1, 2, \ldots, m, \tag{7}$$

where K is a constant tuning parameter. The *ℓ'th global constraint search function* $\tilde{s}_\ell(x)$ is defined as

$$\tilde{s}_\ell(x) = \tilde{s}_{\ell,i}(x), \qquad \forall x \in \Delta_i, \quad \ell = 1, 2, \ldots, m. \tag{8}$$

Note that, by Property 3 above,

$$\tilde{s}_\ell(x) = \min_{i=1,\ldots,E} \tilde{s}_{\ell,i}(x), \quad \forall x \in \Delta.$$

Remark 1 The (single) constant K is a tuning parameter that specifies the trade-off between global exploration, for large K, and local refinement, for small K, when (simultaneously) exploring both the shape of the objective function, via the search function s given in Definition 5, and the extent of the feasible domain, via the constraint search functions $\tilde{s}_\ell$ given in Definition 6.

2.2 Feasible constraint projections

The *feasible constraint projection* process (developed in Section 4 of [12] for linearly constrained domains), when applied to a Delaunay optimization algorithm, ensures that the maximum circumradius of the Delaunay triangulation of the datapoints remains bounded as the iterations proceed, thus leading to better-behaved uncertainty functions near the boundary of the feasible domain.

In this paper, since the search domain $\Omega_s = \{x | a \leq x \leq b\}$ is a simple bound domain, implementation of the feasible constraint projection process is simpler than in [12], as described below.

We first define some preliminary concepts.

Definition 7 A finite set of points $S^k \subset \Omega_s = \{ x \mid a \leq x \leq b \}$ is called *well-situated* [12] with a factor of $r > 1$ if, for any point $x_k \in S^k$ and for all constraints $c^T x \leq d$ of the search domain Ω_s which are not active at x_k, a point $z \in S^k$ lies on the hyperplane $c^T x = d$ such that

$$\frac{\|x_k - z\|}{\|x_k - x'\|} \leq r, \tag{9}$$

where x' is the projection of x_k on the hyperplane $c^T x = d$.

We now present the feasible constraint projection algorithm, developed in Sect. 4 of [12], for the simple bound domain $\Omega_s = \{ x \mid a \leq x \leq b \}$.

Definition 8 Consider $x_k \in \Omega_s$, and S^k as a set in Ω_s that is well-situated with factor r. A *feasible constraint projection* is the iterative adjustment of the point x_k in Ω_s until the resulting augmented set, $S^{k+1} = S^k \cup \{x_k\}$, is also well-situated with factor r. This projection may be achieved with Algorithm 1.

Algorithm 1 Feasible constraint projection of x_k prior to appending to S^k.

1: If $S^{k+1} = S^k \cup \{x_k\}$ is well situated with factor r (see Definition 7), exit.
2: Otherwise, there is a constraint $c^T x = d$ that is not active at x_k, such that there is no point $z \in S^k$ that lies on the hyperplane $c^T x = d$ for which (9) is satisfied. In this case, redefine x_k as the projection of x_k on the hyperplane $c^T x = d$, and repeat from step 1.

In this paper, the value $r = 2$ is found to be suitable for the feasible constraint projection process in our numerical simulations. A detailed explanation of the above algorithm, and proofs of the following properties of the result, are given in [12].

Algorithm 2 Scheme for solving (1) with constant K

1: Set $k = 0$ and initialize S^0 with all of the vertices of Ω_s together with the user-defined initial points (if any are provided). Evaluate $f(x)$ and $c_\ell(x), \forall\, \ell \in \{1, 2, \ldots, m\}$, for all $x \in S^0$.
2: Calculate (or, for $k > 0$, update) interpolating functions $p^k(x)$ and $g_\ell^k(x)$ for the evaluations of $f(x)$ and $c_\ell(x)$, respectively, at all $x \in S^k$.
3: Calculate (or, for $k > 0$, update) a Delaunay triangulation Δ^k over all of the points in S^k.
4: Determine $\hat{x}_k$ as the solution of the following optimization problem:

$$\min_{x \in \Omega_s} \; s^k(x) = p^k(x) - K e^k(x) \tag{10a}$$

$$\text{subject to} \;\; \tilde{s}_\ell^k(x) = g_\ell^k(x) - K e^k(x) \le 0 \quad \forall\, \ell \in \{1, 2, \ldots, m\}. \tag{10b}$$

where the global search function $s^k(x)$ and the ℓ'th global constraint search function $\tilde{s}_\ell^k(x)$ are introduced in Definitions 5 and 6.
5: Define x_k as the feasible constraint projection of $\hat{x}_k$ (Algorithm 1).
6: Evaluate $f(x_k)$ and $c_\ell(x_k)$ $\forall\, \ell \in \{1, 2, \ldots, m\}$ and set $S^{k+1} = S^k \cup \{x_k\}$. Increment k and repeat from step 2.

Property 4 *Consider $R_{\max}$ as the maximum circumradius of a Delaunay triangulation of a set $S \subset \Omega_s = \{x \,|\, a \le x \le b\}$ that is well-situated with factor r. Define $D = \max_{x,y \in \Omega_s} \|x - y\|$ as the diameter of the feasible domain, and $d = \min_{1 \le i \le n}\{b_i - a_i\}$. Then,*

$$R_{\max} \le D\, r_1^{n-1} \quad \textit{where} \quad r_1 = \max\left\{r, \frac{D}{d}\right\}.$$

Property 5 *Consider x' as the feasible constraint projection of x. Then,*

$$\min_{z \in S} \|z - x\| \le \rho \min_{z' \in S} \|z' - x'\|, \quad \rho = \left[2\, r_1^2 \left(1 - \frac{1}{r^2}\right)\right]^{-\frac{n-1}{2}}, \quad r_1 = \max\left\{r, \frac{D}{d}\right\}.$$

That is, if the projected point x' is close to some point $z' \in S$, then the original point x is correspondingly close to $z \in S$ as well.

Remark 2 Since the maximum circumradius $R_{\max}$ is bounded for all iterations, the Lipschitz constant of $e(x)$ is bounded by some corresponding value L_e.

2.3 A Delaunay-based constant *K* algorithm for solving (1)

A Delaunay-based constant K algorithm for solving (1), dubbed Δ-DOGS(Ω), is presented in Algorithm 2.

Figure 1 illustrates the application of Algorithm 2 to a representative $n = 2$ example, in which the objective function $f(x) = x_2$ is minimized within the search domain $\Omega_s = \{x | 0 \le x_1 \le 1, 0 \le x_2 \le 1\}$ subject to $c(x) = x_2 + 0.8\, \sin(x_1 \pi) = 0$ (that is, $c_1(x) = c(x) \le 0$ and $c_2(x) = -c(x) \le 0$), which defines the constraint domain Ω_c as a 1D curve. Note that the feasible region $\Omega := \Omega_c \cap \Omega_s$ is nonconvex.

At each iteration k, by minimizing the search function $s^k(x)$ within the approximation of the feasible domain defined by the $\tilde{s}_\ell^k(x)$ (e.g., within the shaded regions in Fig. 1), the objective function and the extent of the feasible domain are explored simultaneously

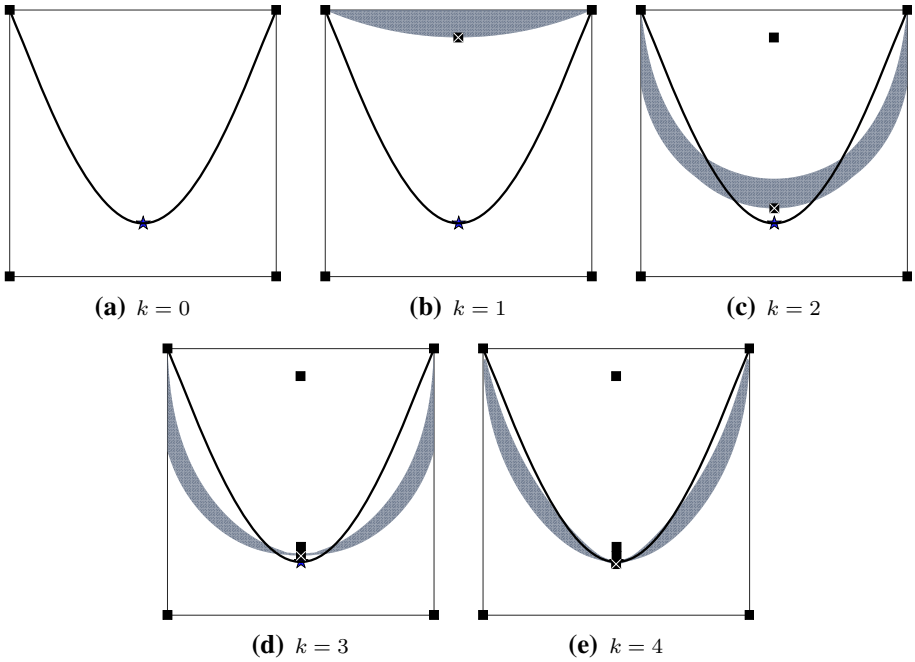


Fig. 1 Illustration of Algorithm 2 on a representative example (see text). The search domain Ω_s is the 2D square indicated, and the global optimum is illustrated by the star. The feasible domain Ω is the 1D curve shown in black. An approximation of Ω (illustrated by the shaded regions) is developed from the available datapoints (black squares) at each iteration k, and is refined by the optimization algorithm as the iterations proceed

using analogous interpolating functions. The solution of (10) in step 4 of Algorithm 2 at each iteration leads to improved estimates of both the objective function and the constraint functions, leading quickly to the minimum of the objective function within the actual feasible domain.

Remark 3 Algorithm 2 is not listed with a specific stopping criterion. It is shown in Sect. 3 that a limit point of the dataset obtained by this algorithm is a solution of the original problem (1). A practical stopping criterion is easily added such that Algorithm 2 terminates after a finite number of iterations. A convenient stopping criterion is $\delta^k \leq \delta_{\text{desired}}$, where $\delta^k = \min_{z \in S^k} \|z - x_k\|$. It follows from the compactness of the search domain Ω_s and the Bolzano–Weierstrass theorem that such termination will occur after a finite number of iterations (see, e.g., Remark 3 in [11]). It is shown later in this paper (see Lemma 2) that a point $z \in S^k$ is computed via this approach such that the residual of the objective function value, $|f(z) - f(x^*)|$, as well as the worst-case constraint function violation, $\max\{c_\ell(z), 0\}$ for all ℓ, are both bounded by δ^k times a prefactor related to the Lipschitz bounds on $f(x)$ and $c_\ell(x)$, which can thus both be made as small as desired by appropriate selection of δ_{desired}.

Step 4 of Algorithm 2 computes the global minimizer of the search function within the current approximation of the feasible domain. In the next subsection, we consider this important step in greater detail.

2.4 Minimizing the search function inside the approximated feasible domain

At step 4 of Algorithm 2, at each iteration k, the global search function $s^k(x)$ is minimized inside the approximation of the feasible domain via solution of (10).

Recall that, within each simplex Δ_i^k in the triangulation Δ^k, the local search function and the local constraint search functions are defined as $s_i^k(x) = p^k(x) - Ke_i^k(x)$ and $\tilde{s}_{\ell,i}^k(x) = g_\ell^k(x) - Ke_i^k(x)$, respectively. Define v_i^k as follows:

$$v_i^k = \text{argmin}_{x\in\Delta_i^k}\ s_i^k(x) \quad \text{subject to} \quad \tilde{s}_{\ell,i}^k(x) \le 0 \quad \forall\, \ell \in \{1,\dots,m\}. \tag{11a}$$

The outcome x_k of step 4 of Algorithm 2 is thus computed as follows:

$$v^k = v_{i_{\min}}^k \quad \text{where} \quad i_{\min} = \text{argmin}_{i\in\{1,\dots,E^k\}}\left[s_i^k(v_i^k)\right], \tag{11b}$$

where E^k is the number of simplices in the triangulation Δ^k. To find v^k in each simplex, we must solve E^k constrained optimization problems (11a), with the nonlinear constraints $\tilde{s}_{\ell,i}^k(x) \le 0$ and the linear constraints $x \in \Delta_i^k$. This computational task is simplified significantly by Lemma 1.

Lemma 1 *If the linear constraints $x \in \Delta_i^k$ in the optimization problems defined in (11a) are relaxed to the entire search domain $x \in \Omega_s$, the resulting values of the optimal points remain unchanged.*

Proof Define u_j^k and u^k [cf. (11)] as

$$u_j^k = \text{argmin}_{x\in\Omega_s}\ s_j^k(x) \quad \text{subject to} \quad \tilde{s}_{\ell,j}^k(x) \le 0 \quad \forall\, \ell \in \{1,\dots,m\}, \quad \forall j, \tag{12a}$$

$$u^k = u_{j_{\min}}^k \quad \text{where} \quad j_{\min} = \text{argmin}_{j\in\{1,\dots,E^k\}}\left[s_j^k(u_j^k)\right]. \tag{12b}$$

We now show that u^k is also a solution of the optimization problem (10). By construction, $u_{j_{\min}}^k = u^k$. According to Property 3 of the uncertainty function, and the fact that $K > 0$,

$$s^k(u^k) \le s_{j_{\min}}^k(u^k), \quad \tilde{s}_\ell^k(u^k) \le \tilde{s}_{\ell,j_{\min}}^k(u^k) \le 0 \quad \forall \ell \in \{1,\dots,m\}. \tag{13}$$

Thus, u^k is a feasible point for optimization problem (10). We now check its optimality; that is, $\forall y \in \Omega_s$ such that $\tilde{s}_\ell^k(y) \le 0 \ \ \forall\, \ell$, that $s^k(u^k) \le s^k(y)$. Assuming that $y \in \Delta_q^k$, by (12) and Property 3 of the uncertainty function,

$$\tilde{s}_{\ell,q}^k(y) = \tilde{s}_\ell^k(y) \le 0. \tag{14}$$

Thus, y is a feasible point for optimization problem (12a). By construction of u^k, $s_{j_{\min}}^k(u^k) \le s_q^k(u_q^k)$, and thus

$$s^k(u^k) \le s_{j_{\min}}^k(u^k) \le s_q^k(u_q^k) \le s_q^k(y) = s^k(y), \tag{15}$$

and the optimality condition is established. □

Remark 4 Lemma 1 shows that the (possibly, multiple) global solutions of (11) and (12) are identical. The process of solving these two problems might lead to different solutions. Note that we just need to find *a* global solution of (11) [or, equivalently (12)] as the algorithm proceeds, so this difference is inconsequential.

Remark 5 If the approximation of the feasible domain at iteration k in problem (10) is empty, then the feasible domain of the subproblem (12a) is empty for all $j \in \{1, \ldots, E^k\}$. In this case, using a derivative-based method like SQP, we can instead find a local minimum of the following objective function:

$$\widehat{\tilde{s}}^k(x) = \sum_{\ell=1}^{m} \max\left\{\tilde{s}_\ell^k(x),\ 0\right\}. \tag{16}$$

At all steps that the approximation of the feasible domain is empty, x_k is taken as the minimizer of the above function in order to search for a feasible point.

The solution of (10) can thus be obtained by solving the optimization problem (12a) for each $j \in \{1, \ldots, E^k\}$ (and, by Lemma 1, for $x \in \Omega_s$). These optimization problems may be solved efficiently using standard derivative-based NLP solvers. Filter SQP [17], SNOPT [20], and IPOPT [57] are among the best derivative-based optimization algorithms available today for such nonlinear programming problems. In our implementation of Algorithm 2, both Filter SQP and SNOPT have been implemented. The initial point which is used when solving (12a) for each $j \in \{1, \ldots, E^k\}$ is taken simply as the body center of simplex j. One of the advantages of using such off-the-shelf SQP-based algorithms for this subproblem is that they can verify, at least locally, whether or not the feasible domain of each subproblem (12a) is empty. If it is, then these solvers find the x that minimizes

$$\widehat{\tilde{s}}_j^k(x) = \sum_{\ell=1}^{m} \max\left\{\tilde{s}_{\ell,j}^k(x),\ 0\right\}.$$

If all of the (12a) subproblems are found to be infeasible, then the resulting values of $\widehat{\tilde{s}}_j^k(x)$ are compared in order to find the x minimizing (16) in the search for a feasible point, as desired, thereby ignoring the search function until a feasible region is identified.

2.5 Parallel implementation

The parallelization approach suggested in Algorithm 5 of [12] extends immediately to the present optimization framework in order to solve (1) in parallel on n_p processors. Note that the three most expensive steps of Algorithm 2 of the present work are as follows:

1. Evaluating the objective and constraint functions. This is assumed to be the most expensive part of the present problem; thus, a framework for simultaneously evaluating the objective and constraint functions at n_p different points of interest on a parallel computer architecture is the focus of this section.
2. Solving the optimization problem (12a) at each simplex via an SQP method. This part of the optimization algorithm is already "embarrassingly parallel", as each subproblem $j \in \{1, \ldots, E^k\}$ may be solved independently.
3. Partitioning the search domain into a Delaunay triangulation. An incremental method is used to update the Delaunay triangulation at each iteration, thus reducing the computational cost of this procedure somewhat. Regardless, the cost of this step increases quickly as the dimension of the problem is increased.

In Algorithm 2, x_k is derived by solving the optimization subproblem (10), then performing a feasible constraint projection. Note, however, that the uncertainty function $e^k(x)$ is independent of the interpolation function $p^k(x)$. Thus, we can calculate $e^{k+i}(x)$, for $i = 1, \ldots, n_p - 1$,

Algorithm 3 Modification of Algorithm 2 such that, at each iteration k, n_p points are identified for parallel evaluation of the objective and constraint functions.

1: Set $k = 0$ and initialize S^0 with all of the vertices of Ω_s together with the user-defined initial points. Evaluate (in parallel) $f(x)$ and $c_\ell(x)$, $\forall\, \ell \in \{1, 2, \ldots, m\}$, for all $x \in S^0$.
2: Calculate (or, for $k > 0$, update) interpolating functions $p^k(x)$ and $g_\ell^k(x)$ for the evaluations of $f(x)$ and $c_\ell(x)$, respectively, at all $x \in S^k$.
3: Calculate (or, for $k > 0$, update) a Delaunay triangulation Δ^k over all of the points in S^k.
4: Determine $\hat{x}_{k,0}$ as a global minimizer of $s^k(x) = p^k(x) - K e^k(x)$ subject to $\tilde{s}_\ell^{k,0}(x) = g_\ell^k(x) - K e^k(x) \leq 0, \forall\, \ell \in \{1, \ldots, m\}$, as in step 4 of Algorithm 2. Note that this calculation may be done in parallel for each simplex. Define $x_{k,0}$ as the feasible constraint projection of $\hat{x}_{k,0}$ (Algorithm 1), and take $S^{k,1} = S^k \cup \{x_{k,0}\}$. Compute $\delta^{k,0} = \min_{y \in S^k} \|x_{k,0} - y\|$.
5: For each substep $i \in \{1, 2, \ldots, n_p - 1\}$, do the following:
   a. Update the Delaunay triangulation of the datapoints in $S^{k,i}$, thus defining the new uncertainty function $e^{k,i}(x)$.
   b. Determine $\hat{x}_{k,i}$ as a global minimizer of $s^{k,i}(x) = p^k(x) - K e^{k,i}(x)$ subject to $\tilde{s}_\ell^{k,i}(x) = g_\ell^k(x) - K e^{k,i}(x) \leq 0, \forall\, \ell \in \{1, \ldots, m\}$. Compute $\delta^{k,i} = \min_{y \in S^{k,i}} \|\hat{x}_{k,i} - y\|$.
   c. If $\delta^{k,i} \leq c\, \delta^{k,0}$ for some c such that $0 < c \leq 1$, replace $\hat{x}_{k,i}$ with a global minimizer of $e^{k,i}(x)$.
   d. Define $x_{k,i}$ as the feasible constraint projection of $\hat{x}_{k,i}$, and take $S^{k,i+1}(x) = S^{k,i} \cup \{x_{k,i}\}$.
6: Take $S^{k+1} = S^{k,n_p}$, and evaluate the objective and constraint functions, $f(x)$ and $c_\ell(x)$, at all of the points $\{x_{k,0}, x_{k,1}, \ldots, x_{k,n_p-1}\}$ in parallel.
7: Increment k, and repeat from step 2 until $\delta^{k,0} \leq \delta_{des}$.

before completing the objective and constraint function evaluations at x_k. That is, steps $k+i$ of Algorithm 2, for $i = 1, \ldots, n_p - 1$, can be performed in parallel with step k under the simplifying assumption that

$$p^{k+i}(x) = p^k(x), \quad \text{and} \quad g_\ell^{k+i}(x) = g_\ell^k(x) \quad \text{for } 1 \leq \ell \leq m. \tag{17}$$

For the purpose of parallelization, we thus impose (17), for $i = 1, \ldots, n_p - 1$ additional iterations, in order to determine, based on the information available at step k, the best $(n_p - 1)$ additional points to explore in parallel with x_k.

Algorithm 3 illustrates how this idea for parallel implementation may be implemented. Note that minimizing $s^{k,i}(x)$ for $0 < i \leq n_p$ is relatively easy, since $s^{k,i}(x) = s^{k,i-1}(x)$ in most of the simplices, and the incremental update of the Delaunay triangulation can be used to flag the indices of those simplices that have been changed by adding $x^{k,i}$ to $S^{k,i-1}(x)$ (see [59]).

3 Convergence analysis

This section analyzes the convergence properties of Algorithm 2. The analysis presented is analogous to, but somewhat different from, that which is presented in §4 of [12]. The convergence analysis is based on the following assumptions.

Assumption 1 The objective and constraint functions $f(x)$ and $c_\ell(x)$, as well as the interpolating functions $p^k(x)$ and $g_\ell^k(x)$ for all k, are Lipschitz for the same Lipschitz constant $\bar{L}$.

Assumption 2 There is a constant $\hat{K}$ such that, for all $x \in \Omega_s$ and $k > 0$,

$$\nabla^2\{p^k(x) - f(x)\} + 2\,\hat{K} I \geq 0, \quad \nabla^2\{g_\ell^k(x) - c_\ell(x)\} + 2\,\hat{K} I \geq 0,$$

$$\nabla^2 p(x) - 2\,\hat{K}\,I \le 0, \quad \nabla^2 g_\ell(x) - 2\,\hat{K}\,I \le 0,$$
$$\nabla^2 f(x) - 2\,\hat{K}\,I \le 0, \quad \nabla^2 c_\ell(x) - 2\,\hat{K}\,I \le 0.$$

Above assumptions are valid for if above functions are twice continuous differentiable.

Assumption 3 The problem in (1) has a nonempty feasible domain, $\Omega \neq \emptyset$. Moreover, since Ω is compact, there exists a minimizer of $f(x)$ in Ω, which is denoted in this section as x^*. (In §5, we relax this assumption to consider the case for which the feasible domain Ω may be empty.)

Lemma 2 *At each step of Algorithm 2, if $K \ge \hat{K}$, then there is a point $\tilde{x} \in \Omega_s$ for which*

$$s^k(\tilde{x}) \le f(x^*) \quad \textit{and} \quad \tilde{s}^k_\ell(\tilde{x}) \le 0 \quad \textit{for } 1 \le \ell \le m, \tag{18}$$

where x^* is a global minimizer of $f(x)$ in Ω.

Proof Consider Δ^k_i as a simplex in Δ^k which includes x^*. Define the functions $F(x)$ and $C_\ell(x)$, $\forall \ell \in \{1, \ldots, m\}$, such that

$$F(x) = p^k(x) - K\,e^k_i(x) - f(x), \quad C_\ell(x) = g^k_\ell(x) - K\,e^k_i(x) - c_\ell(x), \tag{19}$$

where $e^k_i(x)$ is the local uncertainty function in simplex Δ^k_i. Property 2 of the uncertainty function states that $\nabla^2 e^k_i(x) = -2\,I$. Taking the Hessian of (19), we have

$$\nabla^2 F(x) = \nabla^2 p^k(x) - \nabla^2 f(x) + 2\,K\,I, \quad \nabla^2 C_\ell(x) = \nabla^2 g^k_\ell(x) - \nabla^2 c_\ell(x) + 2\,K\,I.$$

By choosing $K > \hat{K}$, according to Assumption 2, $\nabla^2 F(x)$ and $\nabla^2 C_\ell(x)$ are positive semidefinite; thus, $F(x)$ and $C_\ell(x)$ are convex inside the closed simplex Δ^k_i, which includes x^*. Thus, the maximum value of $F(x)$ is located at one of the vertices of Δ^k_i (see, e.g. Theorem 1 of [24]). Moreover, by construction, the value of $F(x)$ and $C_\ell(x)$ at the vertices Δ^k_i are zero; consequently, $F(x^*) \le 0$ and $C_\ell(x^*) \le 0$. On the other hand, $s^k(x^*) = s^k_i(x^*)$, and $\tilde{s}^k_\ell(x^*) = \tilde{s}^k_{\ell,i}(x^*)$. Therefore, $s^k(x^*) \le f(x^*)$ and $g^k_\ell(x^*) - K\,e^k(x^*) \le c_\ell(x^*) \le 0$. □

Remark 6 Lemma 2 shows that the constrained feasible domain is nonempty if $K > \hat{K}$.

Lemma 3 *At each step of Algorithm 2, if $K \ge \hat{K}$, then there is a point $z \in S^k$, such that*

$$f(z) - f(x^*) \le (\bar{L} + K\,L_e)\,\rho \min_{z \in S^k} \|z - x_k\|, \tag{20a}$$

$$c_\ell(z) \le (\bar{L} + K\,L_e)\,\rho \min_{z \in S^k} \|z - x_k\| \quad \forall \ell \in \{1, \ldots, m\}, \tag{20b}$$

where the parameter ρ is defined in Property 5, and is related to the feasible constraint projection procedure.

Proof Choose a point $y \in S^k$ which minimizes $\delta = \min_{y \in S^k} \|y - \hat{x}_k\|$. According to Property 1 of the uncertainty function and Assumption 1, the following inequalities hold, as in the proof of Lemma 3 in [11]:

$$|p^k(\hat{x}_k) - p^k(y)| \le \bar{L}\delta, \quad |g^k_\ell(\hat{x}_k) - g^k_\ell(y)| \le \bar{L}\delta \quad \forall \ell \in \{1, 2, \ldots, m\}, \tag{21a}$$

$$|e^k(\hat{x}_k) - e^k(y)| \le L_e\delta. \tag{21b}$$

Recall that $s^k(x) = p^k(x) - K\,e^k(x)$ and $\tilde{s}^k_\ell(x) = g^k_\ell(x) - K\,e^k(x)$. Using (21),

$$|s^k(y) - s^k(\hat{x}_k)| \le (\bar{L} + K\,L_e)\,\delta, \quad |\tilde{s}^k_\ell(y) - \tilde{s}^k_\ell(\hat{x}_k)| \le (\bar{L} + K\,L_e)\,\delta,$$

$$s^k(y) = p^k(y) = f(y), \quad \tilde{s}_\ell^k(y) = g_\ell^k(y) = c_\ell(y),$$
$$f(y) \le s^k(\hat{x}_k) + (\bar{L} + K L_e)\,\delta, \quad c_\ell(y) \le \tilde{s}_\ell^k(\hat{x}_k) + (\bar{L} + K L_e)\,\delta. \tag{22}$$

Since $\hat{x}_k$ is a global minimizer of $s^k(x)$ with respect to $\tilde{s}_\ell^k(x) \le 0$, it follows from Lemma 2 that $s^k(\hat{x}_k) \le f(x^*)$ and $\tilde{s}_\ell^k(\hat{x}_k) \le 0$, and thus

$$f(y) \le f(x^*) + (\bar{L} + K\,L_e)\,\delta, \quad c_\ell(y) \le (\bar{L} + K\,L_e)\,\delta. \tag{23}$$

In addition, $\delta \le \rho \min_{y \in S^k} \|y - x_k\|$ holds according to Property 5 of the feasible constraint projection. Hence, we have shown that (20) is true for $z = y$. □

Theorem 1 *There is an ω-limit point of the series $\{x_k\}$ which is a global solution of the optimization problem (1).*

Proof Define $T(x) = \max\{f(x) - f(x^*), c_\ell(x)\}$. Take z^k as the value of $x \in S^k$ that minimizes $T(x)$. By construction, $T(x) \ge 0$; thus, $T(z^k) \ge 0$, and $T(z^k)$ is non-increasing with k. Since the search domain is compact, by the Bolzano–Weierstrass theorem, the series $\{x_k\}$ has an ω-limit point. Thus, for any $\varepsilon > 0$, for sufficiently large k, there are i and j such that $i < j \le k$ and $\|x_i - x_j\| \le \varepsilon$. Using Lemma 3 and considering that $z^k \in S^k$, we have

$$0 \le T(z^k) \le (2\,\bar{L} + K\,L_e)\rho\,\varepsilon$$

The above equation is true for all positive values of ε; additionally, since $T(z^k)$ is a non-increasing series; then,

$$\lim_{k \to \infty} T(z^k) = 0.$$

Now define z_1 as an ω-limit point for the z^k. By construction, $T(x)$ is a continuous function of x, which leads immediately to $T(z_1) = 0$. Thus, z_1 is a solution of (1). □

4 Adaptive *K* algorithm

The tuning parameter K in Algorithm 2 specifies the trade-off between global exploration (for large K) and local refinement (for small K). In this section, we develop a method to adjust this tuning parameter at each iteration to maximally accelerate local refinement while still assuring convergence to the global solution of the constrained problem.

The method proposed builds on the fact that, if for each k there exists an $\tilde{x}$ such that $p^k(\tilde{x}) - K\,e^k(\tilde{x}) \le f(x^*)$ subject to $g_\ell^k(\tilde{x}) - K\,e^k(\tilde{x}) \le 0$ for all $\ell \in \{1, 2, \ldots, m\}$, then (18) is satisfied, which is sufficient to establish the convergence of Algorithm 2 in Theorem 1. It is not necessary to choose a constant value for K in Algorithm 2; instead, we may adapt it at each iteration k, taking K^k as bounded and nonnegative with $p^k(\tilde{x}) - K^k\,e^k(\tilde{x}) \le f(x^*)$ subject to $g_\ell^k(\tilde{x}) - K^k\,e^k(\tilde{x}) \le 0$ for all $\ell \in \{1, 2, \ldots, m\}$ at each iteration k.

Assumption 4 The lower bound for the objective function (y_0) over the feasible domain Ω can be estimated.

Take y_0 as a (known) lower bound for $f(x)$ over the feasible domain Ω. By choosing K^k adaptively at each iteration of Algorithm 2 such that

$$0 \le K^k \le K_{\max}, \tag{24a}$$
$$\exists\, \tilde{x} \in \Omega \quad p^k(\tilde{x}) - K^k\,e(\tilde{x}) \le y_0 \text{ and } g_\ell^k(\tilde{x}) \le K^k\,e^k(\tilde{x}), \tag{24b}$$

Algorithm 4 Scheme for solving (1) with adaptive K.

This algorithm is almost identical to Algorithm 2. Instead of solving the subproblem (10) at step 4, the following search function is minimized instead:

$$s_a^k(x) = \max\left\{\frac{p^k(x) - y_0}{e^k(x)}, \frac{g_1^k(x)}{e^k(x)}, \frac{g_2^k(x)}{e^k(x)}, \ldots, \frac{g_m^k(x)}{e^k(x)}\right\}. \tag{25}$$

Note that if $s_a^k(x) \leq 0$, then the following subproblem is solved, which is equivalent to (10) when $K = 0$:

$$\min_{x \in \Omega_s} \; p^k(x) \quad \text{subject to} \quad g_\ell^k(x) \leq 0 \quad \forall \ell = \{1, \ldots, m\}. \tag{26}$$

for all $\ell \in \{1, 2, \ldots, m\}$, this variant of Algorithm 2, with K^k adapted at each iteration k as described above, preserves the guaranteed convergence of the original Algorithm 2 as established in Theorem 1.

Remark 7 If such $K_{\max}$ exists such that K^k satisfies (24a) then the convergence of 4 is trivial. However, since such a $K_{\max}$ does not necessarily exist. More complicated convergence analysis is required which is presented in the rest of this section.

An adaptive K variant of Algorithm 2, for solving (1) when a lower bound y_0 for the objective function $f(x)$ is available, is given in Algorithm 4. Note that reduced values of K^k accelerate local convergence. Thus, at each iteration k of Algorithm 4, we seek the smallest value of K^k which satisfies (24). The optimal K^k is thus taken as

$$K^k = \min_{x \in \Omega_s} s_a^k(x), \tag{27}$$

where $s_a^k(x)$ is defined in (25). It is straightforward to verify that the x that minimizes (27) also minimizes the corresponding search function $p^k(x) - K^k e^k(x)$ subject to $g_\ell^k(x) \leq K^k e^k(x)$ $\forall \ell \in \{1, 2, \ldots, m\}$.

Note that if at some iteration k the solution of (27) is negative, we set $K^k = 0$, and thus the search at iteration k reduces to (26).

Since $e^k(x)$ is defined in a piecewise fashion, to minimize $s_a^k(x)$ in Ω_s, we must solve several optimization problems with linear constraints. Using similar reasoning as in Lemma 1, we can relax these linear constraints.

Again, to minimize $s_{a,i}^k(x)$ within each simplex Δ_i^k, a good initial point is required. Within Δ_i^k, a minimizer of $s_{a,i}^k(x)$ generally has a large value of $e_i^k(x)$; thus, the projection of the circumcenter of Δ_i^k onto the simplex itself provides a reasonable initialization point for the search for the minimum of $s_{a,i}^k(x)$.

The minimization of $s_a^k(x)$ is a minimax problem, akin to those studied in [30,45,47]. In our implementation, we use the exponential fitting method to solve this minimax problem, as explained in detail in [45]. To apply this method, the gradient and Hessian of $(p^k(x) - y_0)/e^k(x)$ and $g_\ell^k(x)/e^k(x)$ are needed. Analytical expressions for these quantities are derived in Sect. 4 in [12].

We now analyze the convergence properties of Algorithm 4 under the same set of assumptions as used in Sect. 3.

Note that the formal proof of convergence of Algorithm 4, given below, is not trivial. The main challenge is that the K^k derived by minimizing $s_a(x)$ is not necessarily bounded. In fact, if $y_0 < f(x^*)$, the value of K^k will go to infinity as the algorithm proceeds. Regardless, Algorithm 4 still converges to the global minimum, as established below.

Theorem 2 *Assuming $y_0 \le f(x^*)$, at each iteration k of Algorithm 4,*

$$\min_{z\in S^k} \max\{f(z) - f(x^*), c_\ell(z)\} \le C\,[\,\rho\,\delta'_k + \sqrt{\rho\,\delta'_k} + \sqrt[4]{\rho\delta'_k}\,], \tag{28}$$

where $\delta'_k = \min_{z\in S^k} \|z - \hat{x}_k\|$, ρ is a parameter defined in Property 5 related to the feasible boundary projection process, $A = \hat{K}\,L_e + \bar{L}$, $B = (f(x^*) - y_0)\,L_e\,\bar{L}^2$, and $C = 2\,\max\{A, B, \sqrt{A}, \sqrt{B}, \sqrt[4]{A}, \sqrt[4]{B}\}$.

Proof We first show that there is a $z \in S^k$ such that

$$\min_{z\in S^k} \max\{f(z) - f(x^*), c_\ell(z)\} \le C\,[\delta_k + \sqrt{\delta_k} + \sqrt[4]{\delta_k}], \tag{29}$$

where $\delta_k = \min_{x\in S^k} \|x - \hat{x}_k\|$. By construction, there are two cases for $\hat{x}_k$.

In the first case, $\hat{x}_k$ is found by solving (26). By construction, $p^k(\hat{x}_k) \le y_0$ and $g_\ell(\hat{x}_k) \le 0$. Now take y as a point in S^k that minimizes $\|\hat{x}_k - y\|$; since $y_0 \le f(x^*)$ and $y \in S^k$, noting Assumptions 1, 2, it follows that

$$f(y) - f(x^*) \le \bar{L}\,\delta_k \quad \text{and} \quad c_\ell(y) \le \bar{L}\,\delta_k, \tag{30}$$

which establishes that (29) is true in this case.

In the second case, $\hat{x}_k$ is found by solving (25). As $\hat{x}_k$ is the minimizer of $s_a^k(x)$, it follows that $s_a^k(\hat{x}_k) \le s_a^k(x^*)$. There are now two possible situations for $s_a^k(x^*)$. In the first situation, $s_a^k(x^*) = g_\ell^k(x^*) \,/\, e^k(x^*)$. Define y and z as the closest points in S^k to $\hat{x}_k$ and x^*, respectively. In Lemma 2, it is shown that $g_\ell^k(x^*) - \hat{K}\,e^k(x^*) - c_\ell(x^*) \le 0$; thus, noting that x^* is feasible, we have

$$\hat{K} \ge g_\ell^k(x^*) \,/\, e^k(x^*). \tag{31}$$

Via Assumption 1 and the fact that $y \in S^k$,

$$\begin{aligned} p^k(y) &= f(y), \qquad g_\ell^k(y) = c_\ell(y), \\ p^k(y) - p^k(\hat{x}_k) &\le \bar{L}\,\delta_k, \qquad g_\ell^k(y) - g_\ell^k(\hat{x}_k) \le \bar{L}\,\delta_k. \end{aligned}$$

In Lemma 2, it is also shown that $p^k(\hat{x}_k) - \hat{K}\,e^k(\hat{x}_k) - f(x^*) \le 0$; thus, using the above equations and $g_\ell^k(\hat{x}_k) \le \hat{K}\,e^k(\hat{x}_k)$, we have

$$f(y) - f(x^*) \le \hat{K}\,e^k(\hat{x}_k) + \bar{L}\,\delta_k, \qquad c_\ell(y) \le \hat{K}\,e^k(\hat{x}_k) + \bar{L}\,\delta_k.$$

By Property 1 of the uncertainty function, and the fact that $e^k(y) = 0$, it follows that $e^k(\hat{x}_k) \le L_e\,\delta_k$, which establishes that (29) is true in this situation. In the situation which is left to analyze, $s_a^k(x^*) = (p^k(x^*) - y_0)/\,e^k(x^*)$. Recall that w is the closet point to x^* in S^k. By Assumptions 1, 2, and the fact that x^* is a feasible point for problem (1), it follows that

$$\begin{aligned} p^k(w) - p^k(\hat{x}_k) \le \bar{L}\,\|w - \hat{x}_k\|, &\qquad g_\ell^k(y) - g_\ell^k(\hat{x}_k) \le \bar{L}\,\|w - \hat{x}_k\|, \\ p^k(w) = f(w), &\qquad g_\ell^k(w) = c_\ell(w), \\ f(w) - f(x^*) \le \bar{L}\,\|w - x^*\|, &\qquad c_\ell^k(w) \le \bar{L}\,\|w - x^*\|. \end{aligned} \tag{32}$$

By Lemma 4 in [11], we have

$$\|w - x^*\|^2 \le e^k(x^*). \tag{33}$$

Using (33) and the square of (32) leads to

$$\Big(f(w) - f(x^*)\Big)^2 \le \bar{L}^2 e^k(x^*), \quad c_\ell(z)^2 \le \bar{L}^2\,e^k(x^*),$$

$$\max\left\{\Big(f(w)-f(x^*)\Big)^2, c_\ell(w)^2\right\} \le \bar{L}^2\, e^k(x^*). \tag{34}$$

Since $\hat{x}_k$ is a minimizer of $s_a^k(\hat{x}_k) \le s_a^k(x^*)$, and $s_a^k(x^*) = (p^k(x^*)-y_0)/\,e^k(x^*)$,

$$\frac{p^k(\hat{x}_k)-y_0}{e^k(\hat{x}_k)} \le \frac{p^k(x^*)-y_0}{e^k(x^*)}, \qquad \frac{g_\ell^k(\hat{x}_k)}{e^k(\hat{x}_k)} \le \frac{p^k(x^*)-y_0}{e^k(x^*)}.$$

Thus,

$$\frac{p^k(\hat{x}_k)-y_0}{e^k(\hat{x}_k)} \le \frac{p^k(x^*)-f(x^*)}{e^k(x^*)} + \frac{f(x^*)-y_0}{e^k(x^*)}, \tag{35}$$

$$\frac{g_\ell^k(\hat{x}_k)}{e^k(\hat{x}_k)} \le \frac{p^k(x^*)-f(x^*)}{e^k(x^*)} + \frac{f(x^*)-y_0}{e^k(x^*)}. \tag{36}$$

As in (31), we can show that

$$\hat{K} \ge p^k(x^*) - f(x^*)\,/\,e^k(x^*). \tag{37}$$

Using (35), (36), and (37)

$$\frac{p^k(\hat{x}_k)-y_0}{e^k(\hat{x}_k)} \le \hat{K} + \frac{f(x^*)-y_0}{e^k(x^*)}, \qquad \frac{g_\ell^k(\hat{x}_k)}{e^k(\hat{x}_k)} \le \hat{K} + \frac{f(x^*)-y_0}{e^k(x^*)},$$

$$\frac{p^k(\hat{x}_k)-f(x^*)}{e^k(\hat{x}_k)} \le \hat{K} + \frac{f(x^*)-y_0}{e^k(x^*)}, \qquad \frac{g_\ell^k(\hat{x}_k)}{e^k(\hat{x}_k)} \le \hat{K} + \frac{f(x^*)-y_0}{e^k(x^*)},$$

$$f(y) - \bar{L}\,\delta_k \le p^k(\hat{x}_k), \qquad c_\ell(y) - \bar{L}\,\delta_k \le g_\ell^k(\hat{x}_k),$$

$$\frac{f(y)-\bar{L}\,\delta_k - f(x^*)}{e^k(\hat{x}_k)} \le \hat{K} + \frac{f(x^*)-y_0}{e^k(x^*)},$$

$$\frac{c_\ell(y)-\bar{L}\,\delta_k}{e^k(\hat{x}_k)} \le \hat{K} + \frac{f(x^*)-y_0}{e^k(x^*)}.$$

We thus conclude that

$$\max\{f(y)-f(x^*), c(y)\} \le \Big(\hat{K} + \frac{f(x^*)-y_0}{e^k(x^*)}\Big)e^k(\hat{x}_k) + \bar{L}\,\delta_k.$$

According to Property 1 and the fact that $y \in S^k$,

$$\max\{f(y)-f(x^*), c(y)\} \le \Big(\hat{K} + \frac{f(x^*)-y_0}{e^k(x^*)}\Big)L_e\,\delta_k + \bar{L}\,\delta_k. \tag{38}$$

Define the variables ϕ_w and ϕ_y as follows:

$$\phi_w = \max\left\{\Big(f(w)-f(x^*)\Big)^2, c_\ell^k(w)^2\right\}, \quad \phi_y = \max\left\{f(y)-f(x^*),\ c_\ell(y)\right\}.$$

Using (34) and (38),

$$\phi_y\,\phi_w \le \Big(\hat{K}\,L_e + \bar{L}\Big)\,\delta_k\,\phi_w + \Big(f(x^*)-y_0\Big)\,L_e\,\bar{L}^2\,\delta_k,$$

$$\phi_y\,\phi_w \le A\,\delta_k\,\phi_w + B\,\delta_k.$$

Defining $u = \max\{\phi_y\,,\,\phi_w\}$ and $v = \min\{\phi_y\,,\,\phi_w\}$, it follows that[2]

$$v \le 2\,\max\{A\,\delta_k, \sqrt{B\,\delta_k}\} \le 2\,A\,\delta_k + 2\sqrt{B\,\delta_k}. \tag{39}$$

[2] If $A, B, C > 0$ and $A^2 \le A\,B + C$ then $A \le B + \sqrt{C} \le 2\,\max\{B, \sqrt{C}\}$.

If $v = \phi_y$, then (29) is a direct outcome of (39); otherwise,

$$\max\left\{\Big(f(w) - f(x^*)\Big), c_\ell^k(w)\right\} \le C\,[\sqrt{\delta_k} + \sqrt[4]{\delta_k}], \tag{40}$$

which establishes that (29) is true in this situation as well. Thus, (29) is valid for all cases. By Property 5, $\delta_k \le \rho\|x - x_k\|$; thus, (28) is obtained from (29). □

Remark 8 By Theorem 2 above, as in Theorem 1, we can easily show that, if $y_0 \le f(x^*)$, an ω-limit point of the datapoints determined by Algorithm 4 is a solution of (1).

4.1 Using an inaccurate estimate of y_0

In the previous section, convergence of Algorithm 4 is proved when $y_0 \le f(x^*)$. It is observed (see Sect. 6) that, if y_0 is not a tight lower bound for the global minimum, the rate of convergence is reduced. In this subsection, we study the behavior of Algorithm 4, when the estimated value of y_0 is somewhat *larger* than the actual minimum of the function of interest within the feasible domain. It is shown that, upon convergence, Algorithm 4 determines a feasible point z such that $f(z) \le y_0$.

Theorem 3 *Assuming $y_0 > f(x^*)$, at each step of Algorithm 2, there is a point $z \in S^k$ such that*

$$\max\{f(z) - y_0, c_\ell(z)\} \le [\bar{L} + \hat{K}\,L_e]\,\rho\,\delta_k, \quad \delta_k = \min_{z\in S^k}\|x_k - z\|. \tag{41}$$

Proof As in Theorem 2, we first show that

$$\max\{f(y) - y_0,\ c_\ell(y)\} \le [\bar{L} + \hat{K}\,L_e]\|y - \hat{x}_k\|, \tag{42}$$

where $y \in S^k$ minimizes $\delta_k = \|z - \hat{x}_k\|$. As before, during the iterations of Algorithm 4, there are two possible cases for step k. In the first case, $\hat{x}_k$ is found by solving (26). Similar to the first case in Theorem 2,

$$\begin{aligned}
&p^k(y) - p^k(\hat{x}_k) \le \bar{L}\delta_k, \quad f(y) - p^k(\hat{x}_k) \le \bar{L}\delta_k,\\
&g_\ell^k(y) - g_\ell^k(\hat{x}_k) \le \bar{L}\delta_k, \quad c_\ell(y) - g_\ell^k(\hat{x}_k) \le \bar{L}\,\delta_k,\\
&\max\{f(y) - y_0, c_\ell(y)\} \le \bar{L}\,\delta_k,
\end{aligned}$$

which establishes that (42) is true in this case.

In the second case, $\hat{x}_k$ is found by solving (25), and is a minimizer of $s_k^a(x)$. As in (31), using the fact that $y_0 > f(x^*)$, it is easy to show that

$$\begin{aligned}
&\max\left\{p^k(x^*) - f(x^*),\ g_\ell^k(x^*)\right\} \le \hat{K}e^k(x^*),\\
&s_a(x^*) = \frac{\max\{p^k(x^*) - y_0,\, g_\ell^k(x^*)\}}{e^k(x^*)} \le \hat{K}.
\end{aligned}$$

Since $\hat{x}_k$ is a global minimizer of $s_a^k(x)$, it follows that

$$\begin{aligned}
&s_a^k(\hat{x}_k) \le s_a^k(x^*) \le \hat{K},\\
&\max\left\{p^k(\hat{x}_k) - y_0,\ g_\ell^k(\hat{x}_k)\right\} \le \hat{K}\,e^k(\hat{x}_k) \le \hat{K}\,L_e\,\|y - \hat{x}_k\|, &(43)\\
&p^k(y) - p^k(\hat{x}_k) \le L_e\,\delta_k, \quad g_\ell^k(y) - g_\ell^k(\hat{x}_k) \le L_e\,\delta_k, &(44)\\
&p^k(y) = f(y), \quad g_\ell^k(y) = c_\ell(y). &(45)
\end{aligned}$$

Using (43), (44) and (45), then (42) is satisfied. Finally, using Lemma 2 as in Theorem 2, (41) is obtained from (42). □

Remark 9 Analogous to Remark 8 for the case with $y_0 \leq f(x^*)$, it follows easily from Theorem 3 above, for the case with $y_0 > f(x^*)$, that an ω-limit point of the datapoints determined by Algorithm 4 is a feasible point of (1) with objective function value less than or equal to y_0.

5 The case of an empty feasible domain

In the previous section, it was assumed that the feasible domain Ω of the problem considered, (1), is nonempty. We now consider the behavior of Algorithm 2 when the feasible domain of (1) might be empty (and, thus, Assumption 3 might not hold), though Assumptions 1, 2 remain in effect. We will show that Algorithm 2 can be used, in fact, to *verify* whether or not the feasible domain is empty.

Lemma 4 *If the feasible domain of (1) is empty, and $K > \hat{K}$; then,*

$$\sum_{\ell=1}^{m} \max\left\{c_\ell(y), 0\right\} \leq \sum_{\ell=1}^{m} \max\left\{c_\ell(x_f),\ 0\right\} + m\,(\bar{L} + K\,L_e)\,\rho \min_{z \in S^k} \|z - x_k\|, \quad (46)$$

where x_f is the point $x \in \Omega_s$ that globally minimizes $\sum_{\ell=1}^{m} \max\{c_\ell(x)\,, 0\}$, and y is the point $z \in S^k$ that minimizes $\|z - \hat{x}_k\|$.

Proof Take Δ_i^k as a simplex in Δ^k which includes x_f. In the proof of Lemma 2, it is shown that $C_\ell(x_f) \leq 0$ if $K \geq \hat{K}$, where $C_\ell(x)$ is defined in (19); thus,

$$\begin{gathered} g_\ell(x_f) - K\,e^k(x_f) \leq c_\ell(x_f), \\ \sum_{l=1}^{m} \max\left\{g_\ell(x_f) - K\,e^k(x_f),\ 0\right\} \leq \sum_{\ell=1}^{m} \max\left\{c_\ell(x_f),\ 0\right\}. \end{gathered} \quad (47)$$

By construction, either $\hat{x}_k$ is a feasible point of (10), or it is a minimizer of $\sum_{\ell=1}^{m} \max\{g_\ell(x) - K\,e^k(x), 0\}$. In either case, we have:

$$\sum_{\ell=1}^{m} \max\left\{g_\ell(\hat{x}_k) - K\,e^k(\hat{x}_k),\ 0\right\} \leq \sum_{\ell=1}^{m} \max\left\{g_\ell(x_f) - K\,e^k(x_f),\ 0\right\}.$$

Using (47),

$$\sum_{\ell=1}^{m} \max\left\{g_\ell(\hat{x}_k),\ 0\right\} \leq m\,K\,e^k(\hat{x}_k) + \sum_{\ell=1}^{m} \max\left\{c_\ell(x_f),\ 0\right\}.$$

By construction, L_e and $\bar{L}$ are Lipschitz norms for $e^k(x)$ and $g_\ell(x)$, respectively. Moreover, $\rho\,\delta_k \leq \|y - \hat{x}_k\|$. Using Assumption (3), we have

$$\sum_{\ell=1}^{m} \max\left\{g_\ell^k(y),\ 0\right\} \leq \sum_{\ell=1}^{m} \max\left\{c_\ell(x_f),\ 0\right\} + m\,(\bar{L} + K\,L_e)\,\rho\,\delta_k.$$

Since $y \in S^k$, it follows that $g_\ell^k(y) = c_\ell(y)$, and thus (46) is satisfied. □

Remark 10 By Lemma 4 above (cf. Theorem 1), if Algorithm 2 is not terminated at any step, an ω-limit point of the datapoints determined is a global minimizer of $\sum_{\ell=1}^{m} \max\{c_\ell(x), 0\}$ in Ω_s. If Algorithm 2 is terminated at step k, an approximation of this point is obtained; this approximation is improved as k is increased.

6 Results

Several benchmark optimizations are now considered to study the behavior of the algorithms developed in this work.

The test problems considered in this work considered, each defined such that $f(x^*) = 0$, are:

(A) A simple linear objective function, defined over an n-dimensional space, subject to a nonlinear equality constraint, generated using a Rastrigin function, defining an $(n-1)$-dimensional nonconvex feasible domain:

$$\min_{x\in\Omega_s} \quad f(x_1, \ldots, x_n) = x_n - 0.1, \quad \text{subject to} \quad c(x) = 0, \tag{A.1}$$

$$c(x) = x_n - \frac{1}{12}\sum_{i=1}^{n-1}\left\{3.5^2\,(x_i - 0.7)^2 - 2\,\cos\Big(7\pi\,(x_i - 0.7)\Big)\right\} - (n-1)/6 - 0.1, \tag{A.2}$$

$$0 \le x_1, x_2, \ldots, x_n \le 1. \tag{A.3}$$

This problem has 4^{n-1} local minima, including the unique global minimum $x^* = [0.7, \ldots, 0.7, 0.1]^T$, with $f(x^*) = 0$.

(B) A quadratic objective function (given by the distance to the origin), defined over an n-dimensional space, subject to a nonlinear inequality constraint, again generated using a Rastrigin function, defining a disconnected feasible domain characterized by 2^n distinct "islands" within the search domain:

$$\min_{x\in\Omega_s} \quad f(x) = x^T\,x - 0.024\,n, \quad \text{subject to} \quad c(x) \le 0, \tag{B.1}$$

$$c(x) = \frac{n}{12} + \frac{1}{6}\sum_{i=1}^{n}\left\{4\,(x_i - 0.7)^2 - 2\,\cos\Big(4\pi\,(x_i - 0.7)\Big)\right\}, \tag{B.2}$$

$$0 \le x_1, x_2, \ldots, x_n \le 1. \tag{B.3}$$

This problem has 2^n local minima, including the unique global minimum $x^* = [0.154969, 0.154969, \ldots, 0.154969]^T$, with $f(x^*) = 0$.

(C) A quadratic objective function (given by the distance to the origin), defined over a 2D space, subject to two nonlinear inequality constraints defining a nonconvex feasible domain with a "petal"-shaped hole (see Simionescu [53]):

$$\min_{x\in\Omega_s} \quad f(x_1, x_2) = x_1^2 + x_2^2 - 0.64, \quad \text{subject to} \quad -1 \le c(x) \le 0, \tag{C.1}$$

$$c(x) = \left\{r_t + r_s\cos\left[n_s\,\tan^{-1}\left(\frac{x_1}{x_2}\right)\right]\right\}^2 - x_1^2 - x_2^2, \tag{C.2}$$

$$-1.25 \le x_1, x_2 \le 1.25, \tag{C.3}$$

where $r_t = 1$, $r_s = 0.2$, and $n_s = 8$. This problem has eight global minima, located at $x = [\pm 0.306, \pm 0.739]^T$ and $x = [\pm 0.739, \pm 0.306]^T$, each characterized by $f(x^*)$=0.

(D) A linear objective function, defined over an 2-dimensional space, subject to a sinusoidal and quadratic inequality constraints, defining an $(n-1)$-dimensional nonconvex feasible domain, (see LSQ problem in [23]):

$$\min_{x\in\Omega_s} \quad f(x_1,x_2) = x_1 + x_2 - 0.6, \quad \text{subject to} \quad c_1(x) \le 0, \quad c_2 \le 0, \tag{D.1}$$

$$c_1(x) = \frac{3}{2} - x_1 - 2\,x_2 - \frac{1}{2}\sin\Big(2\pi\,(x_1^2 - 2x_2)\Big) \tag{D.2}$$

$$c_2(x) = x_1^2 + x_2^2 - \frac{3}{2}, \tag{D.3}$$

$$0 \le x_1, x_2 \le 1. \tag{D.4}$$

Unlike [23] we treat the $f(x)$ in the LSQ problem as unknown. But similar to citegramacy2016modeling the search domain is defined by a simple bound domain, and the constraint domain is define with sinusoidal and quadratic inequality functions ($c_1(x)$ and $c_2(x)$). This problem has a global minima, located at $x = [0.195, 0.40]^T$ and characterized by $f(x^*) \approx 0$.

In the following, we present application of Δ-DOGS(Ω) on a test problem whose objective function is nonconvex.

(E) GSBP which is presented in [23] and can be formulated as follows:

$$\min_{x} \quad f(x_1,x_2), \quad \text{subject to} \quad c_1(x) \le 0, \quad c_2(x) = 0, \quad c_3(x) = 0. \tag{E.1}$$

$$\begin{aligned}
f(x_1,x_2) &= \frac{\log\Big((1+a)\,(30+b)\Big) - 8.69}{2.43} + 0.4897, \quad \text{with}\\
a &= (4\,x_1 + 4\,x_2 - 3)^2\,(75 - 56\,(x_1+x_2) + 3\,(4\,x_1-2)^2\\
&\quad + 6\,(4\,x_1-2)\,(4\,x_2-2) + 3\,(4\,x_2-2)^2),\\
b &= (8\,x_1 - 12\,x_2 + 2)^2\,\Big(-14 - 128\,x_1 + 12\,(4\,x_1-2)^2\\
&\quad + 192\,x_2 - 36\,(4\,x_1-2)\,(4\,x_2-2) + 27\,(4\,x_2-2)^2\Big),
\end{aligned} \tag{E.2}$$

$$c_1(x) = \frac{3}{2} - x_1 - 2\,x_2 - \frac{1}{2}\sin\Big(2\pi\,(x_1^2 - 2x_2)\Big), \tag{E.3}$$

$$\begin{aligned}
c_2(x) &= 15 - \Big(15\,x_2 - \frac{5}{4\,\pi^2}\,(15\,x_1 - 5)^2\\
&\quad + \frac{5}{\pi}\,(15\,x_1 - 5) - 6\Big)^2 - 10\,\Big(1 - \frac{1}{8\,\pi}\Big)\,\cos(15\,x_1 - 5),
\end{aligned} \tag{E.4}$$

$$\begin{aligned}
c_3(x) &= 4 - \Big(4 - 2.1\,(2\,x_1-1)^2 + \frac{(2\,x_1-1)^4}{3}\Big)\,(2\,x_1-1)^2\\
&\quad - (2\,x_1-1)^2\,(2\,x_2-1)^2 - 16\,(x_2^2 - x_2)\,(2\,x_2-1)^2\\
&\quad - 3\,\sin[12\,(1-x_1)] - 3\,\sin[12\,(1-x_2)],
\end{aligned} \tag{E.5}$$

$$0 \le x_1, x_2 \le 1. \tag{E.6}$$

Problem E is a 2D benchmark problem in which $f(x)$ is the Goldstein-Price function (rescaled and centered) [43], the search domain is a simple bound domain, the constraint domain is mix of inequality and equality non-linear functions where $c_1(x)$ is sinusoidal, $c_2(x)$ is the Branin function [25], and $c_3(x)$ is a nonlinear function defined in [44]. This problem has a global minima located at $x = [0.95, 0.467]^T$ characterized by $f(x^*) \approx 0$.

(F) G3: Nonconvex objective function with nonlinear equality constraint.

$$\min_{x\in\Omega_s} \quad f(x) = 1 - (\sqrt{n})^n \, \Pi_{i=1}^n x_i, \quad \text{subject to} \quad c(x) = 0, \tag{F.1}$$

$$c(x) = \sum_{i=1}^{n} x_i^2 - 1, \tag{F.2}$$

$$0 \le x_i \le 1, \tag{F.3}$$

This problem has a global minima, located at $x^* = [1/n^{0.5}, \ldots, 1/n^{0.5}]^T$ and characterized by $f(x^*) = 0$.

(G) G6: Nonconvex objective function with nonlinear inequality constraint functions.

$$\min_{x\in\Omega_s} \quad f(x_1, x_2), \quad \text{subject to} \quad c_1(x) \le 0, \quad c_2 \le 0, \tag{G.1}$$

$$f(x_1, x_2) = \sum_{i=1}^{2} (a_i + (100 - a_i)\, x_i - r_i)^3 + 6961.81388, \tag{G.2}$$

$$c_1(x) = -\sum_{i=1}^{2} (a_i + (100 - a_i)\, x_i - b_i)^2 + 100, \tag{G.3}$$

$$c_2(x) = \sum_{i=1}^{2} (a_i + (100 - a_i)\, x_i - d_i)^2 - 82.81, \tag{G.4}$$

$$0 \le x_i \le 1. \tag{G.5}$$

where $r = [10, 20]^T$, $b = [5, 5]^T$, $d = [6, 5]^T$, and $a = [13, 0]^T$. This problem has a global minima, located at $x^* = [0.0126, 0.0084]^T$ and characterized by $f(x^*) \approx 0$. This function has an interesting property in which the variation of the objective function, $f(x)$ close to the global minimizer is higher in x_2 direction compared with x_1 direction. We developed a surrogate function in [4] named multivariant adaptive polyharmonic spline (MAPS) for such problems. However, to have fair comparison for all methods in this paper we use natural polyharmonic spline [58] as a surrogate function.

(H) G8: nonlinear objective function with nonlinear inequality constraint functions.

$$\min_{x\in\Omega_s} \quad f(x_1, x_2) \quad \text{subject to} \quad c_1(x) \le 0, \quad c_2 \le 0, \tag{H.1}$$

$$f(x_1, x_2) = 0.095825 - \frac{\sin^3(20\pi x_1)\sin(20\pi x_2)}{x_1^3(x_1 + x_2) \times 10^4}, \tag{H.2}$$

$$c_1(x) = 100\, x_1^2 - 10\, x_2 + 1, \tag{H.3}$$

$$c_1(x) = 1 - 10\, x_1 + (10\, x_2 - 4)^2, \tag{H.4}$$

$$0 < x_i \le 1. \tag{H.5}$$

This problem has a global minima, located at $x^* = [0.122, 0.425]^T$ and characterized by $f(x^*) \approx 0$. The objective function is nonsmooth close to zero; therefore, the lower bound for the search domain is considered greater than zero. In the simulations, we consider the lower bound as 10^{-2}.

In Sect. 6.2, Algorithm 2 (constant K) and the (typically, more practical) Algorithm 4 (adaptive K) are applied to three test problems, and the roles of the tuning parameters K (on Algorithm 2) and y_0 (on Algorithm 4) are studied. Finally, Sect. 6.3 compares the performance

Table 1 Implementation of Algorithms 2 and 4 on Problems A, B, and C, denoting z as the best point found, and $f(z)$ and $c(z)$ as the value of the objective and constraint functions at this point

Prob.	Alg.	Parameter	Converged	$f(z)$	$c(z)$	# of fn. evals.
A	2	$K = 1$	No	0.08	-0.02	13
		$K = 2$	Yes	0	-0.002	16
		$K = 10$		0.007	-0.077	30
	4	$y_0 = 0.9$	No	0.287	0.005	11
		$y_0 = 0$	Yes	0.001	-0.001	18
		$y_0 = -0.02$		0.005	-0.006	25
B	2	$K = 5$	No	0.347	0.006	11
		$K = 12$	Yes	0.002	-0.015	25
		$K = 15$		0.002	-0.015	42
	4	$y_0 = 0.752$	No	0.335	-0.035	18
		$y_0 = 0$	Yes	0.002	-0.0203	23
		$y_0 = -0.028$		-0.003	0.032	45
C	2	$K = 0$	No	0.183	0.005	7
		$K = 5$	Yes	-0.01	-0.034	16
		$K = 10$		0.016	0.011	30
	4	$y_0 = 0.16$	No	0.152	0.133	15
		$y_0 = 0$	Yes	0.01	-0.013	22
		$y_0 = -0.04$		-0.01	-0.034	40

Algorithm 2 is tested on each problem three times, once with $K < \hat{K}$, once with $K \approx \hat{K}$, and once with $K > \hat{K}$, where $\hat{K}$ is the value of K required to achieve convergence. Algorithm 4 is also tested on each problem three times, once with $y_0 > f(x^*)$, once with $y_0 = f(x^*)$, and once with $y_0 < f(x^*)$; note that all three of these benchmark problems are constructed with $f(x^*) = 0$

of Algorithm 4 with other modern derivative-free optimization methods on a representative test problem with nonconvex constraints, assuming accurate knowledge of y_0.

In the test optimizations performed in this section, polyharmonic spline interpolation [58] is used for interpolation of the known values of the objective and constraint functions. The optimizations are stopped, at iteration k, when the new datapoint, x_k, is within a $\delta_{desired}$ neighborhood of an existing datapoint (in S^k); the present simulations take $\delta_{desired} = 0.01$.

6.1 Test problems

To highlight the unique features of the algorithms developed, the three test optimization problems chosen for this study, described below, have nonconvex equality and inequality constraints, in certain cases even defining disconnected feasible domains. To compare the performance of the optimization algorithms in finding a global minimum amongst several local minima, the number of function evaluations required in order to achieve a desired level of convergence is used as the evaluation criterion.

6.2 Illustrative test problems for Algorithms 2 and 4

The performance of Algorithms 2 and 4 on Problems A, B, and C, in $n = 2$ dimensions, and the dependence of convergence on the the tuning parameters K and y_0 are compared in Table 1.

It is observed in Table 1 that Algorithm 2 converges to the global minimum whenever the parameter K is made sufficiently large, and that unnecessarily large values of K result in additional global exploration over the search domain, consequently slowing convergence. Similarly, it is observed that Algorithm 4 converges to the global minimum whenever the parameter $y_0 \leq f(x^*)$, and that unnecessarily small values of y_0 result in additional global exploration over the search domain, again slowing convergence. In cases for which $y_0 > f(x^*)$, a feasible point is identified for which objective function at least as small as y_0.

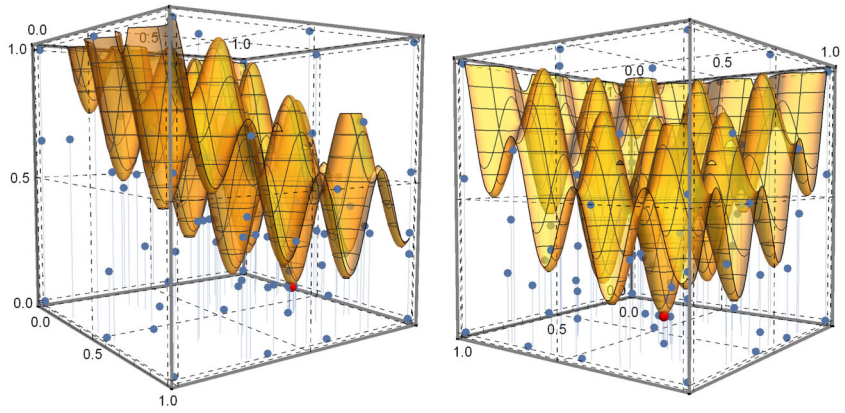
Figure 2 considers the extension of Algorithm 4 to Problems A and B in optimization problems with $n = 3$, taking $y_0 = 0$ in both cases. For Problem A, as in the $n = 2$ the equality constraint (A.1) constrains the feasible domain to an $n - 1 = 2$ dimensional surface. For Problem B, the inequality constraint (B.1) results in a feasible domain with several disconnected "islands" within the search domain. It is seen in Fig. 2 that convergence on both problems, which exhibit complex nonconvex feasibility domains, is remarkably rapid. As illustrated in the figure, upon solution, the best point found (using only 91 function evaluations) in Problem A is $z = \{0.101, 0.704, 0.707\}$, with $f(z) = 0.0012$ and $c(z) = 0.0017$ (the global minimum is $x^* = \{0.1, 0.7, 0.7\}$). The best point found (using only 87 function evaluations) in Problem B is $z = \{0.149, 0.165, 0.153\}$, with $f(z) = 0.0006$ and $c(z) = -0.0065$ (the global minimum is $x^* = \{0.155, 0.155, 0.155\}$).

6.3 Comparison with other derivative-free methods

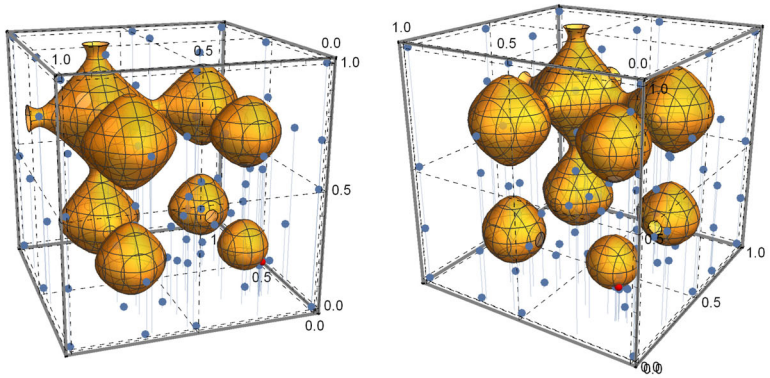
We now briefly compare the new optimization algorithm developed here with some other leading derivative-free optimization schemes on the difficult class of optimization problems considered in this work. We compare the results on Problem B.

The first algorithm considered is the pattern search method called Mesh Adaptive Direct Search (MADS), as proposed in [5]. We apply the efficient implementation of MADS in the NOMAD software package [31]. The NOMAD solver (implementing MADS with $2n$ neighbors) is a local derivative-free optimization algorithm that can solve difficult non-differentiable optimization problems.

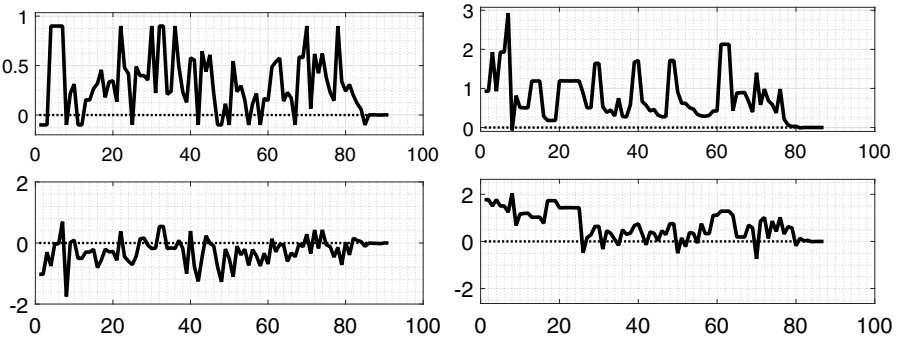
The second method considered is the Surrogate Management Framework (SMF) proposed by [14]. SMF is one of the most popular derivative-free, globally-convergent, computationally-efficient optimization algorithms available today. The SMF algorithm has two main steps, "search" and "poll". SMF is a hybrid method that combines a pattern search with a surrogate-based optimization algorithm, and switches back and forth between (local) polling and (global) searching steps. The surrogate used by SMF is usually developed using a Kriging interpolation method. In practice, the optimization is usually initialized in the SMF approach using an initial set of datapoints generated with a Latin Hypercube Sampling (LHS) approach [14,37,39], which generally provides a well-distributed set of datapoints in a parameter space with box constraints, with each input variable fairly well distributed over its feasible range. The search step then uses (and updates) a Kriging-based surrogate to look for reduced objective function values inside the feasible domain, which is discretized onto a Cartesian grid that is successively refined as the iterations proceed. The search continues until it fails to return a new point on the current grid with a reduced value of the objective function, at which point a pattern search (e.g., such as the one in MADS) is used to poll the neighborhood around the current best point. If the poll step succeeds in finding a point with a reduced objective function value, then the surrogate model is updated and another search is performed; if it does not, the grid is refined and the process repeated. The implementation of



(a) Location of datapoints used in solution of Problem A (two different views). The objective function is a distance from horizontal plane passing $[0, 0]^T$. The yellow regions are the feasible domain and the red dot is a global minimizer.



(b) Location of datapoints used in solution of Problem B (two different views). The objective function is a distance from the origin $[0, 0]^T$. The yellow regions are the feasible domain and the red dot is a global minimizer.



(c) $f(x^k)$ and $c(x^k)$ versus k in Problem A. **(d)** $f(x^k)$ and $c(x^k)$ versus k in Problem B.

Fig. 2 Algorithm 4 applied to the $n = 3$ cases of Problems A and B. (Color figure online)

SMF that is utilized in the present work is the same method used in [49], and was developed by our group in collaboration with Professor Alison Marsden.

The third method considered in this section, for comparison purposes, is the Δ-DOGS(C) algorithm reported previously by our group (see [11]). Δ-DOGS(C) is a highly efficient, provably convergent, nonlinearly-constrained Delaunay-based optimization algorithm, which in many cases compares favorably with SMF in terms of computational efficiency (see [11]). Like the present algorithm, which is also in the Δ-DOGS family, Δ-DOGS(C) leverages a synthetic uncertainty function built on the framework of a Delaunay triangulation of existing datapoints, together with an interpolation of all existing datapoints using any desired well-behaved interpolation strategy (many of our numerical experiments thus far have used polyharmonic spline interpolation).

The difficult class of problems considered in the present work is characterized by expensive constraint functions $c_\ell(x)$, as well as expensive objective functions $f(x)$; both $c_\ell(x)$ and $f(x)$ are probed on the fly as the iterations proceed, and the $c_\ell(x)$ together ultimately define a nonconvex (possibly even disconnected) feasible domain Ω. The global optimization algorithm developed in two different variants (Algorithm 2 with constant K, and Algorithm 4 with adaptive K) in the present work, dubbed Δ-DOGS(Ω), is designed specifically for problems of this class. In contrast, the three comparison methods described above [MADS, SMF, and Δ-DOGS(C)] were each designed to minimize a single nonconvex function inside a known (a priori) convex feasible domain. We thus define the following new objective function in order to apply these three existing schemes to the difficult class of problems considered in this work:

$$\tilde{f}(x) = \max\left\{ f(x) - y_0, \max_{\ell=1,\ldots,m} \{c_\ell(x), 0\} \right\}. \tag{55}$$

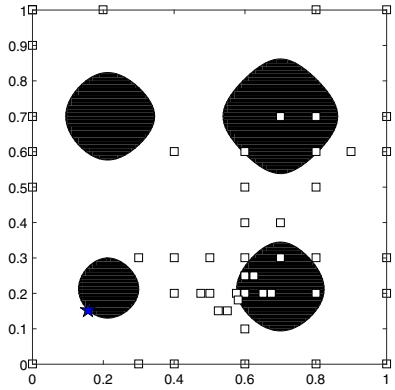
6.3.1 Test problems with convex objective function with conconvex constraints

Problem B is well suited to characterize and compare the four different schemes considered here [that is, to compare MADS, SMF, and Δ-DOGS(C), with the objective function as defined in (55), with Δ-DOGS(Ω)] on this difficult class of problems. In all cases, a stopping criterion of $\delta_{desired} = 0.01$ is applied [see Remark 3]. Note that many derivative-free optimization methods, like MADS, in fact do not guarantee convergence to the global minimum of the problem considered. Furthermore, if the solution of the problem considered is on the boundary of feasibility, as in Problem B, the task of finding a global solution is especially difficult.

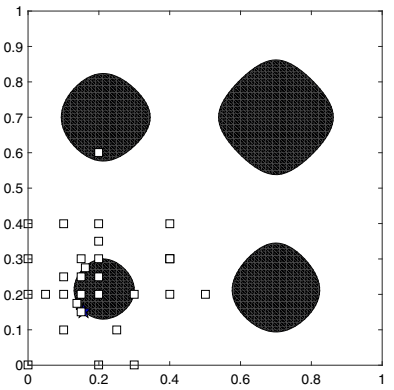
Comparison of the five plots in Fig. 3, and the corresponding data summarized in Tables 2, 3, indicate that the number of function evaluations is minimized in this (typical) example using Algorithm 4, which accurately locates the feasible global minimizer in this case with only 23 in Problem B function evaluations.

The SMF method (Fig. 3c) explores the search domain globally and locates the global minimizer (albeit with significantly reduced precision) with 48 function evaluations. Note that more than half of the function evaluations are performed in the polling steps in order to guarantee convergence. Also note that the datapoints tend to accumulate in the vicinity of the feasible global minimizer as the iterations proceed, thereby causing the Kriging interpolation model to become numerically ill-conditioned. Due to this ill-conditioning of the Kriging method itself, achieving more accurate convergence using Kriging-based SMF proves to be quite difficult.

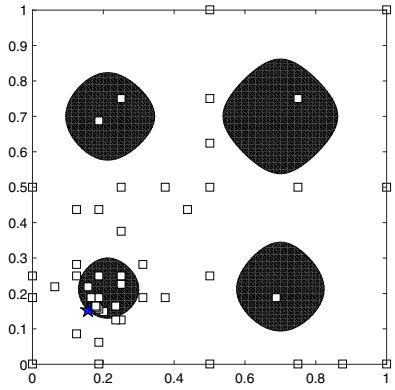
The Δ-DOGS(C) algorithm (Fig. 3d) fails to converge to the global minimizer on this problem; since the objective function (55) is non-differentiable, this method is in fact not



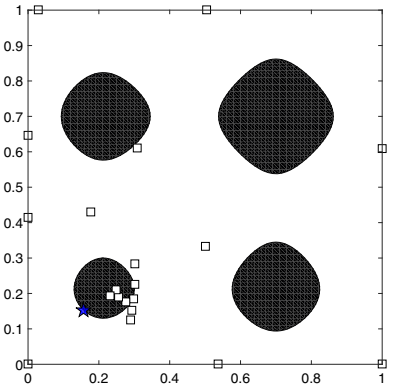
(a) MADS with $x_0 = [1, 0]^T$.



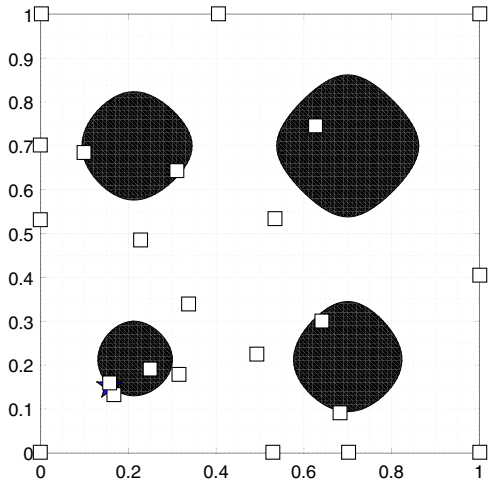
(b) MADS with $x_0 = [0.4, 0.3]^T$.



(c) SMF.



(d) Δ-DOGS(C).



(e) Algorithm 4, also known as Δ-DOGS(Ω).

Fig. 3 Comparison, on Problem B, of MADS (via NOMAD), SMF, and Δ-DOGS(C) with the adaptive K variant of the Δ-DOGS(Ω) algorithm developed and analyzed in this work, as presented in Algorithm 4

Table 2 Performance comparison on Problem B with $n = 2$

Algorithm	Initial point(s)	Converged?	$f(z)$	$c(z)$	# of fn. evals.
MADS	{1, 0}	No	0.328	− 0.011	50
MADS	{0.4, 0.3}	Yes	− 0.003	0.031	31
SMF	$3n + 3$ LHS points	Yes	0.01	− 0.074	48
Δ-DOGS(C)	3 points	No	0.047	− 0.147	24
Algorithm 4	2^n vertices	Yes	0.002	− 0.020	23

Table 3 Performance comparison on Problems A-D with convex objective function and nonconvex constraints with $n = 2$

Problem	Algorithm	Converged?	$f(z)$	$c(z)$	# of fn. evals.
A-2D	SMF	Yes	0.006	0.009	33
	Δ-DOGS(C)	No	0.02	− 0.07	29
	Algorithm 4	Yes	0.001	− 0.001	18
B-2D	SMF	Yes	0.01	− 0.074	48
	Δ-DOGS(C)	No	0.047	− 0.147	24
	Algorithm 4	Yes	0.002	− 0.020	23
C-2D	SMF	Yes	0.007	0.009	32
	Δ-DOGS(C)	Yes	0.005	− 0.005	30
	Algorithm 4	Yes	0.001	− 0.001	22
D-2D	SMF	Yes	0.007	− 0.006	33
	Δ-DOGS(C)	Yes	0.004	− 0.09	28
	Algorithm 4	Yes	0.002	− 0.020	23

guaranteed to converge on this problem. Note, however, that this method does perform global exploration during the search, and successfully locates a feasible point near the boundary of feasibility, which is fairly close to the global minimizer, with only 24 function evaluations. By relaxing the stopping criterion $\delta_{desired} = 0.01$ mentioned previously, it was found that this method would eventually locate the global minimizer using about 40 function evaluations.

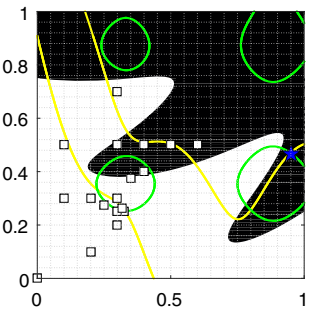
The MADS algorithm (Fig. 3a, b) converges only locally. If a good initial guess is provided (Fig. 3b), the global minimizer is located, to about the same level of accuracy as Algorithm 4, in 31 function evaluations. If a good initial guess is not provided (Fig. 3a), the global minimizer is not located by the time the stopping criterion of $\delta_{desired} = 0.01$ is reached (after 50 function evaluations).

6.3.2 Test problems with nonconvex objective function with nonconvex constraints

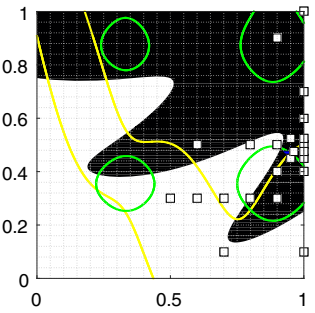
In this part we present application of Δ-DOGS(Ω) on a test problem whose objective function is nonconvex.

Now we apply the presented algorithms (Δ-DOGS(Ω), Δ-DOGS, SMF, MADS) in the Sect. 6.3 on the problem GSBP. Figure 4 illustrates the performance of Algorithm 4 on the problem E, which was able to find the global solution, the blue star marker, after 27 evaluations.

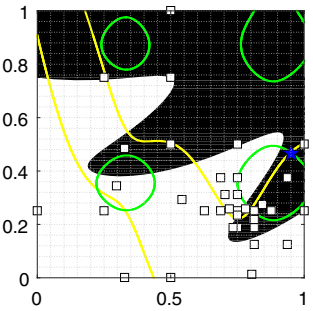
Comparison of the five plots in Fig. 4, and the corresponding data summarized in Tables 4, 5 indicates that the number of function evaluations is minimized in this (typical) example



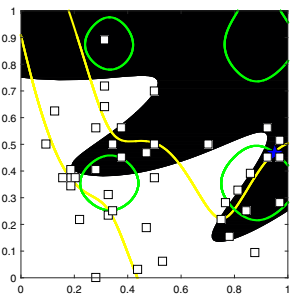
(a) MADS with $x_0 = [0.7, 1]^T$.



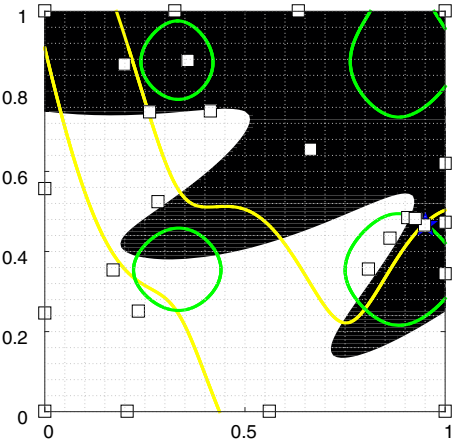
(b) MADS with $x_0 = [0.9, 0.3]^T$.



(c) SMF.



(d) Δ-DOGS(C).



(e) Algorithm 4, also known as Δ-DOGS(Ω) with $y_0 = 0.48$.

Fig. 4 Comparison, on Problem E, of MADS (via NOMAD), SMF, and Δ-DOGS(C) with the adaptive K variant of the Δ-DOGS(Ω) algorithm developed and analyzed in this work, as presented in Algorithm 4

using Algorithm 4, which accurately locates the feasible global minimizer in this case with only 27 in the GSBP problem function evaluations.

The SMF method (Fig. 4c) explores the search domain globally, but converged to a local solution.

Table 4 Performance comparison on GBSP Problem

Algorithm	Initial point(s)	Converged?	$f(z)$	max $c(z)$	# of fn. evals.
MADS	{0.7, 1}	No	0.335	0.039	33
MADS	{0.9, 0.3}	Yes	− 0.483	0.006	28
SMF	9 LHS points	No	0.352	0.411	36
Δ-DOGS(C)	3 points	No	− 0.219	0.593	40
Algorithm 4	4 vertices	Yes	− 0.492	0.057	27

Table 5 Performance comparison on problem E-H with nonconvex objective function and nonlinear constraints with $n = 2$

Problem	Algorithm	Converged?	$f(z)$	$c(z)$	# of fn. evals.
E-2D	SMF	No	0.352	0.411	33
	Δ-DOGS(C)	No	− 0.219	0.593	40
	Algorithm 4	Yes	− 0.492	0.057	27
F-2D	SMF	Yes	0.0013	0.0093	37
	Δ-DOGS(C)	Yes	0.002	0.005	15
	Algorithm 4	Yes	− 0.00	0.0001	9
G-2D	SMF	No	− 18	0.0082	65
	Δ-DOGS(C)	no	− 1011	11	14
	Δ-DOGS w/ MAPS	Yes	− 0.08	0.003	9
	Algorithm 4	Yes	− 0.09	0.002	10
H-2D	SMF	Yes	0.035	0.00	27
	Δ-DOGS(C)	Yes	0.007	0.00	15
	Algorithm 4	Yes	0.09	0.003	10

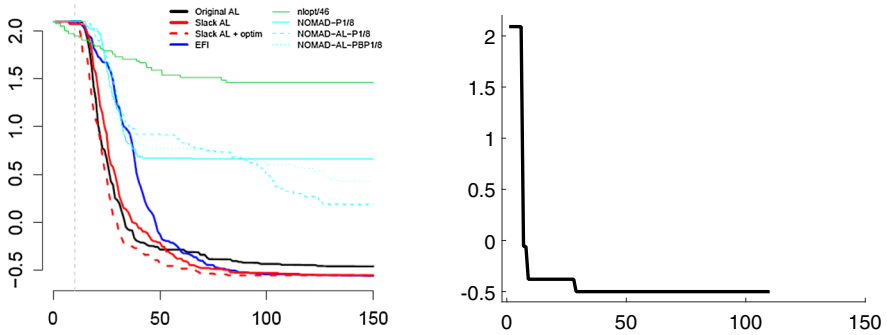
The Δ-DOGS(C) algorithm (Fig. 4d) fails to converge to the global minimizer on this problem as well; since the objective function (55) is non-differentiable, this method is in fact not guaranteed to converge on this problem.

The MADS algorithm (Fig. 4a, b) converges only locally. If a good initial guess is provided (Fig. 4b), the global minimizer is located, to about the same level of accuracy as Algorithm 4, in 28 function evaluations. If a good initial guess is not provided (Fig. 4a), the global minimizer is not located by the time the stopping criterion of $\delta_{desired} = 0.01$ is reached (after 33 function evaluations).

Furthermore, Fig. 5 compares the solution of Algorithm 4 with other schemes reported in [23]. As we see our scheme outperforms the other state of the art methods for nonconvex functions with nonconvex feasible domain.

6.4 Application of Algorithm 4 on the Lockwood problem

We also used the proposed algorithm to solve the Lockwood problem [23,40]. This problems has six variables with a linear objective function and two nonconvex constraint blackbox functions.



(a) Performance of different methods as reported in [23]; figure reprodcued from [23], with permission.

(b) Algorithm 4.

Fig. 5 Results on the GBSP problem

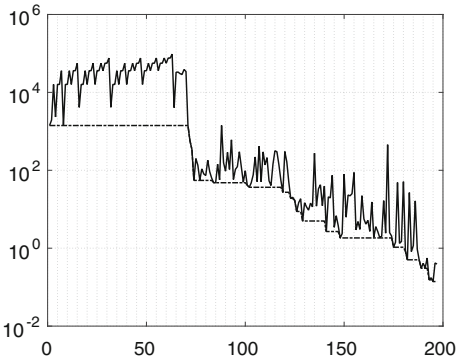


Fig. 6 Lockwood convergence result. The vertical axis shows $\tilde{f}(x)$ as defined in (55) and the horizontal axis shows the number of evaluations. The dashed-line shows the pareto front of the best solution found thus far

In this problem, the objective function is defined as the pumping rates that are proportional to the cost of operating the system, which is minimized subject to the constraints on the contaminant staying within the plume boundaries [23].

Let x_j be the pumping rate for the jth well, the optimization problem is formulated as [23,40]:

$$\text{minimize } f(x) = \sum_{j=1}^{6} x_j \quad \text{subject to} \quad x \in \Omega := \Omega_c \cap \Omega_s \subseteq \mathbb{R}^6 \quad \text{where} \qquad \text{(F)}$$

$$\Omega_c = \{x | c_1(x) \leq 0\,,\, c_2(x) \leq 0\}, \quad \Omega_s = \{x \in \mathbb{R}^6 | \, 0 \leq x_j \leq 2 \times 10^4\},$$

Figure 6 shows the convergence of Δ-DOGS(Ω) to the vicinity of a minimizer of Lockwood problem with $y_0 = 24,000$.

The motivating Lockwood example does not hold the twice differentiality assumption on constraint functions [46]. The feasible points of the search domain are projected on to the boundary of feasibility $\{x \in \mathbb{R}^6 : c(x) = 0\}$ [46], which violates smoothness assumptions (2). Although there is no guarantee of convergence in this problem, the Algorithm 4 still finds a point with the given target value that minimizes the constraint violation.

7 Conclusions

This paper presents a new Delaunay-based derivative-free optimization approach, dubbed Δ-DOGS(Ω), of the same general family as introduced in [11,12]. The approach developed here is designed specifically for the optimization of computationally expensive nonconvex functions within a nonconvex (possibly lower-dimensional, or even disconnected) feasible domain, which is itself defined by computationally expensive constraint functions which are explored as the iterations proceed. Two main variants of Δ-DOGS(Ω) have been presented, both of which are available from the authors upon request.

Algorithm 2 uses any well-behaved interpolation strategy, such as polyharmonic splines, for both the objective function $f(x)$ and the constraint functions $c_\ell(x)$, together with a synthetic piecewise-quadratic uncertainty function built on the framework of a Delaunay triangulation. A search function defined by comparing simple functions of the uncertainty model and the interpolants (of both the objective and the constraint functions) is minimized within the search domain at each iteration, and new objective and constraint function computations are performed at the optimized point, thereby refining the surrogate models of the objective and constraint functions at each iteration until convergence is achieved. Convergence to the feasible global minimum is proved mathematically under reasonable technical conditions on the smoothness of the objective and constraint functions.

Algorithm 4 modifies Algorithm 2 to use an estimate of the lower bound of the function to maximally accelerate local refinement while still ensuring convergence to the global minimizer.

We have also proposed, in Algorithm 3, a framework to efficiently parallelize, on multiple processors, the objective and constraint function evaluations required by Algorithm 2 at each step in the optimization process; this parallelization approach extends in an obvious fashion to the parallelization of Algorithm 4.

There is an inherent "curse of dimensionality" associated with derivative-free optimization problems. High-dimensional derivative-free optimization problems are generally computationally intractable with any method; for high-dimensional optimization problems, derivative-based methods should always be preferred.

The main limitation of the derivative-free optimization algorithms developed in the present work, and the related optimization algorithms in [11,12], is the memory requirements of the Delaunay triangulation algorithms upon which this body of work is based. The required Delaunay triangulations makes the "curse of dimensionality" associated with this family of optimization algorithms even more pronounced than it might be otherwise. Constructing Delaunay triangulations in problems higher than about ten dimensions is generally computationally intractable. The focus in this body of work on the use of Delaunay triangulations, which are generally the most "regular" triangulations possible for a given distribution of datapoints, provides an essential ingredient in our proofs of convergence. Regardless, for practical applications, the notion of using triangulations that are "nearly" Delaunay might be helpful in future work for practically extending the algorithms developed here to somewhat higher-dimensional problems.

Though the test problems considered in this paper illustrate well the key features of the new optimization algorithms presented, in future work we will test these optimization algorithms on additional benchmark and application-focused problems. We will also consider different interpolation approaches as alternatives to the standard polyharmonic spline interpolation approach used here.

Problems in which the feasibility at point x is only computable in a binary fashion (feasible or infeasible), rather than given by the union of inequalities based on computable (and, smooth) constraint functions $c_\ell(x)$, will also be considered. Problems in which both the objective and constraint function evaluations are inaccurate will also be considered.

Acknowledgements The authors gratefully acknowledge Dr. Fred Y. Hadaegh and Dr. Firouz M. Naderi for their support, Professors Alison Marsden for her assistance in developing an efficient SMF code, Robert Gramacy and Sebastien Le Digabel for sharing their Lockwood test problem code, and Dr. Stefan Wild for his constructive feedback. The authors gratefully acknowledge funding from AFOSR FA 9550-12-1-0046, from the Cymer Center for Control Systems & Dynamics, from the Leidos corporation in support of this work. Also, the research was supported by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

References

1. Abramson, M.A., Audet, C., Dennis, J.E., Le Digabel, S.: OrthoMADS: a deterministic MADS instance with orthogonal directions (2009)
2. Achterberg, T.: Scip: solving constraint integer programs. Math. Program. Comput. **1**(1), 1–41 (2009)
3. Alimo, S., Beyhaghi, P., Meneghello, G., Bewley, T.: Delaunay-based optimization in cfd leveraging multivariate adaptive polyharmonic splines (maps). The American Institute of Aeronautics and Astronautics, SciTech Meeting (2017) (2017)
4. Alimohammadi, S., He, D.: Multi-stage algorithm for uncertainty analysis of solar power forecasting. In: Power and Energy Society General Meeting (PESGM), 2016, pp. 1–5. IEEE (2016)
5. Audet, C., Dennis, J.E.: A pattern search filter method for nonlinear programming without derivatives. SIAM J. Optim. **14**(4), 980–1010 (2004)
6. Audet, C., Dennis, J.E.: Mesh adaptive direct search algorithms for constrained optimization. SIAM J. Optim. **17**(1), 188–217 (2006)
7. Audet, C., Dennis, J.E.: A progressive barrier for derivative-free nonlinear programming. SIAM J. Optim. **20**(1), 445–472 (2009)
8. Belitz, P., Bewley, T.: New horizons in sphere-packing theory, part ii: lattice-based derivative-free optimization via global surrogates. J. Glob. Optim. pp. 1–31 (2013)
9. Belotti, P., Lee, J., Liberti, L., Margot, F., Wächter, A.: Branching and bounds tighteningtechniques for non-convex minlp. Optim. Methods Softw. **24**(4–5), 597–634 (2009)
10. Bertsekas, D.P.: On penalty and multiplier methods for constrained minimization. SIAM J. Control Optim. **14**(2), 216–235 (1976)
11. Beyhaghi, P., Bewley, T.R.: Delaunay-based derivative-free optimization via global surrogates, part ii: convex constraints. J. Global Optim. **66**(3), 383–415 (2016)
12. Beyhaghi, P., Cavaglieri, D., Bewley, T.: Delaunay-based derivative-free optimization via global surrogates, part i: linear constraints. J. Glob. Optim. pp. 1–52 (2015)
13. Boissonnat, J.-D., Devillers, O., Hornus, S.: Incremental construction of the delaunay triangulation and the delaunay graph in medium dimension. In: Proceedings of the Twenty-Fifth Annual Symposium on Computational Geometry, pp. 208–216. ACM (2009)
14. Booker, A.J., Dennis Jr., J.E., Frank, P.D., Serafini, D.B., Torczon, V., Trosset, M.W.: A rigorous framework for optimization of expensive functions by surrogates. Struct. Optim. **17**(1), 1–13 (1999)
15. Boyd, S., Vandenberghe, L.: Convex Optimization, vol. 25. Cambridge University Press, Cambridge (2010)
16. Jones, D.R.: A taxonomy of global optimization methods based on response surfaces. J. Global Optim. **21**(4), 39 (2001)
17. Fletcher, R., Leyffer, S., Toint, P.L.: On the global convergence of a filter-SQP algorithm. SIAM J. Optim. **13**(1), 44–59 (2002)
18. Forsgren, A., Gill, P.E.: Primal-dual interior methods for nonconvex nonlinear programming. SIAM J. Optim. **8**(4), 1132–1152 (1998)
19. Forsgren, A., Gill, P.E., Wright, M.H.: Interior Methods for Nonlinear Optimization, vol. 44. SIAM, Philadelphia (2002)
20. Gill, P.E., Murray, W., Saunders, M.A.: Snopt: an sqp algorithm for large-scale constrained optimization. SIAM Rev. **47**(1), 99–131 (2005)

21. Gill, P.E., Murray, W., Saunders, M.A., Wright, M.H.: A Schur-complement method for sparse quadratic programming. Technical report, DTIC Document (1987)
22. Gill, P.E., Saunders, M.A., Wong, E.: On the performance of SQP methods for nonlinear optimization. In: Modeling and Optimization: Theory and Applications, pp. 95–123. Springer, Berlin (2015)
23. Gramacy, R.B., Gray, G.A., Le Digabel, S., Lee, H.K.H., Ranjan, P., Wells, G., Wild, S.M.: Modeling an augmented lagrangian for blackbox constrained optimization. Technometrics **58**(1), 1–11 (2016)
24. Hoffman, K.L.: A method for globally minimizing concave functions over convex sets. Math. Program. **20**(1), 22–32 (1981)
25. Jones, D.R., Perttunen, C.D., Stuckman, B.E.: Lipschitzian optimization without the lipschitz constant. J. Optim. Theory Appl. **79**(1), 157–181 (1993)
26. Jones, D.R., Schonlau, M., Welch, W.J.: Efficient global optimization of expensive black-box functions. J. Global Optim. **13**(4), 455–492 (1998)
27. Kort, B.W., Bertsekas, D.P.: A new penalty function method for constrained minimization. In: Proceedings of the 1972 IEEE Conference on Decision and Control, 1972 and 11th Symposium on Adaptive Processes, pp. 162–166. IEEE, New York (1972)
28. Kort, B.W., Bertsekas, D.P.: Combined primal-dual and penalty methods for convex programming. SIAM J. Control Optim. **14**(2), 268–294 (1976)
29. Krige, D.G.: A statistical approach to some basic mine valuation problems on the Witwatersrand. J. South. Afr. Inst. Min. Metall. **52**, 119–139 (1952)
30. Lawrence, C.T., Zhou, J.L., Tits, A.L.: User's guide for CFSQP version 2.0: AC code for solving (large scale) constrained nonlinear (minimax) optimization problems, generating iterates satisfying all inequality constraints (1994)
31. Le Digabel, S.: Algorithm 909: nomad: nonlinear optimization with the mads algorithm. ACM Trans. Math. Softw. (TOMS) **37**(4), 44 (2011)
32. Lewis, R.M., Torczon, V.: Pattern search algorithms for bound constrained minimization. SIAM J. Optim. **9**(4), 1082–1099 (1999)
33. Lewis, R.M., Torczon, V.: Pattern search methods for linearly constrained minimization. SIAM J. Optim. **10**(3), 917–941 (2000)
34. Lewis, R.M., Torczon, V.: A globally convergent augmented lagrangian pattern search algorithm for optimization with general constraints and simple bounds. SIAM J. Optim. **12**(4), 1075–1089 (2002)
35. Long, C.C., Marsden, A.L., Bazilevs, Y.: Shape optimization of pulsatile ventricular assist devices using FSI to minimize thrombotic risk. Comput. Mech. **54**(4), 921–932 (2014)
36. Madani, R., Ashraphijuo, M., Lavaei, J.: Promises of conic relaxation for contingency-constrained optimal power flow problem. IEEE Trans. Power Syst. **31**(2), 1297–1307 (2016)
37. Marsden, A.L., Feinstein, J.A., Taylor, C.A.: A computational framework for derivative-free optimization of cardiovascular geometries. Comput. Methods Appl. Mech. Eng. **197**(21), 1890–1905 (2008)
38. Marsden, A.L., Wang, M., Dennis Jr., J.E., Moin, P.: Optimal aeroacoustic shape design using the surrogate management framework. Optim. Eng. **5**(2), 235–262 (2004)
39. McKay, M.D., Beckman, R.J., Conover, W.J.: A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. Technometrics **42**(1), 55–61 (2000)
40. Meer, J.T.M.T., Duijne, H.V., Nieuwenhuis, R., Rijnaarts, H.H.M.: Prevention and reduction of pollution of groundwater at contaminated megasites: integrated management strategy, and its application on megasite cases. In: Quevauviller, P. (ed.) Groundwater Science and Policy: An International Overview, pp. 405–420 (2008)
41. Misener, R., Floudas, C.A.: Antigone: algorithms for continuous/integer global optimization of nonlinear equations. J. Glob. Optim. **59**(2–3), 503–526 (2014)
42. Moghadam, M.E., Migliavacca, F., Vignon-Clementel, I.E., Hsia, T.-Y., Marsden, A.L.: Optimization of shunt placement for the norwood surgery using multi-domain modeling. J. Biomech. Eng. **134**(5), 051002 (2012)
43. Molga, Marcin., Smutnicki, Czesław.: Test functions for optimization needs. *Test functions for optimization needs*, 101, (2005)
44. Parr, J.M., Keane, A.J., Forrester, A.I.J., Holden, C.M.E.: Infill sampling criteria for surrogate-based optimization with constraint handling. Eng. Optim. **44**(10), 1147–1166 (2012)
45. Pee, E.Y., Royset, J.O.: On solving large-scale finite minimax problems using exponential smoothing. J. Optim. Theory Appl. **148**(2), 390–421 (2011)
46. 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)
47. Polak, E., Royset, J.O., Womersley, R.S.: Algorithms with adaptive smoothing for finite minimax problems. J. Optim. Theory Appl. **119**(3), 459–484 (2003)

48. Pussoli, B.F., Da Silva, L.W., Barbosa, J.R., Kaviany, M.: Optimization of peripheral finned-tube evaporators using entropy generation minimization. Int. J. Heat Mass Transf. **55**(25–26), 7838–7846 (2012)
49. Ramachandra, A.B., Sankaran, S., Humphrey, J.D., Marsden, A.L.: Computational simulation of the adaptive capacity of vein grafts in response to increased pressure. J. Biomech. Eng. **137**(3), 031009 (2015)
50. Rasmussen, C.E.: Gaussian processes for machine learning. Int. J. Neural Syst. **14**(2), 69–106 (2006)
51. Sacks, J., Welch, W.J., Mitchell, T.J., Wynn, H.P.: Design and analysis of computer experiments. Stat. Sci. **4**(4), 409–423 (1989)
52. Schonlau, M., Welch, W.J., Jones, D.R.: A data-analytic approach to Bayesian global optimization. In: Department of Statistics and Actuarial Science and The Institute for Improvement in Quality and Productivity, 1997 ASA Conference (1997)
53. Simionescu, P.A.: Computer-Aided Graphing and Simulation Tools for AutoCAD Users, vol. 32. CRC Press, Boca Raton (2014)
54. Snoek, J., Larochelle, H., Adams, R.P.: Practical Bayesian optimization of machine learning algorithms. In: Advances in Neural Information Processing Systems, pp. 2951–2959 (2012)
55. Tawarmalani, M., Sahinidis, N.V.: A polyhedral branch-and-cut approach to global optimization. Math. Program. **103**, 225–249 (2005)
56. Torczon, V.: On the convergence of pattern search algorithms. SIAM J. Optim. **7**(1), 1–25 (1997)
57. Wächter, A., Biegler, L.T.: On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Math. Program. **106**(1), 25–57 (2006)
58. Wahba, G.: Spline Models for Observational Data, vol. 59. SIAM, Philadelphia (1990)
59. Watson, D.F.: Computing the n-dimensional delaunay tessellation with application to voronoi polytopes. Comput. J. **24**(2), 167–172 (1981)
60. Wild, S.M., Regis, R.G., Shoemaker, C.A.: Orbit: optimization by radial basis function interpolation in trust-regions. SIAM J. Sci. Comput. **30**(6), 3197–3219 (2008)