Optimization Methods and Software

ISSN: 1055-6788 (Print) 1029-4937 (Online) Journal homepage: https://www.tandfonline.com/loi/goms20

CONORBIT: constrained optimization by radial basis function interpolation in trust regions

Rommel G. Regis & Stefan M. Wild

To cite this article: Rommel G. Regis & Stefan M. Wild (2017) CONORBIT: constrained optimization by radial basis function interpolation in trust regions, Optimization Methods and Software, 32:3, 552-580, DOI: 10.1080/10556788.2016.1226305

To link to this article: https://doi.org/10.1080/10556788.2016.1226305

Optimization Methods & Software, 2017
Vol. 32, No. 3, 552–580, http://dx.doi.org/10.1080/10556788.2016.1226305

Taylor & Francis
Taylor & Francis Group

CONORBIT: constrained optimization by radial basis function interpolation in trust regions†

Rommel G. Regis[a*] and Stefan M. Wild[b]

[a]*Department of Mathematics, Saint Joseph's University, Philadelphia, PA 19131, USA;* [b]*Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL 60439, USA*

(*Received 8 October 2015; accepted 16 August 2016*)

This paper presents CONORBIT (CONstrained Optimization by Radial Basis function Interpolation in Trust regions), a derivative-free algorithm for constrained black-box optimization where the objective and constraint functions are computationally expensive. CONORBIT employs a trust-region framework that uses interpolating radial basis function (RBF) models for the objective and constraint functions, and is an extension of the ORBIT algorithm [S.M. Wild, R.G. Regis and C.A. Shoemaker, *ORBIT: optimization by radial basis function interpolation in trust-regions*, SIAM J. Sci. Comput. 30 (2008), pp. 3197–3219]. It uses a small margin for the RBF constraint models to facilitate the generation of feasible iterates, and extensive numerical tests confirm that such a margin is helpful in improving performance. CONORBIT is compared with other algorithms on 27 test problems, a chemical process optimization problem, and an automotive application. Numerical results show that CONORBIT performs better than COBYLA (Powell 1994), a sequential penalty derivative-free method, an augmented Lagrangian method, a direct search method, and another RBF-based algorithm on the test problems and on the automotive application.

Keywords: constrained optimization; black-box optimization; computationally expensive function; simulation-based optimization; trust region; fully linear model; radial basis function interpolation

AMS Subject Classifications: 65K05; 90C56

1. Introduction

In this paper, we develop a derivative-free algorithm for constrained optimization where the objective and constraint functions are black boxes and computationally expensive. Formally, our goal is to solve the nonlinear optimization problem

$$\min\{f(x) : g(x) \leq 0,\ l \leq x \leq u\}, \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is the objective function, $g : \mathbb{R}^n \to \mathbb{R}^q$ is the vector-valued inequality constraint function, and $l, u \in \mathbb{R}^n$ are the lower and upper bounds on the variables, respectively. We let $g_i(x)$ denote an individual inequality constraint function, and so $g(x) = (g_1(x), \ldots, g_q(x))$.

*Corresponding author. Email: rregis@sju.edu

†The submitted manuscript has been created by UChicago Argonne, LLC, Operator of Argonne National Laboratory ('Argonne'). Argonne, a U.S. Department of Energy Office of Science laboratory, is operated under Contract No. DE-AC02-06CH11357. The U.S. Government retains for itself, and others acting on its behalf, a paid-up, nonexclusive, irrevocable worldwide license in said article to reproduce, prepare derivative works, distribute copies to the public, and perform publicly and display publicly, by or on behalf of the Government.

Here, f and g are black-box functions whose values are outputs of a computationally expensive, but deterministic, simulation and whose derivatives are unavailable.

For convenience, we employ set notation for the constraints. Thus, we let

$$\mathcal{B} = \{x \in \mathbb{R}^n : l \leq x \leq u\} \tag{2}$$

denote the region corresponding to the bound constraints and take

$$\Omega = \{x \in \mathcal{B} : g(x) \leq 0\} \tag{3}$$

to be the feasible region of (1).

We assume the following about the feasible region:

(C1) The set $\mathcal{B}$ is bounded and has a nonempty interior ($-\infty < l < u < \infty$).
(C2) f and g are continuously differentiable on the set $\mathcal{B}$.
(C3) Ω has a nonempty interior.

As a consequence of (C1), Ω is bounded. As a consequence of (C2), the constraints $g(x) \leq 0$ are assumed to be *quantifiable* and *relaxable* on the bounded region $\mathcal{B}$; see [25] for formal definitions of these terms. For now, we assume, through (C3), that there are no black-box equality constraints. Moreover, we assume that a feasible starting point is given. This last assumption is not unreasonable in many engineering design problems, since a practitioner often has an existing, feasible design and is looking for an improved solution. Future work will extend the proposed algorithm to target black-box equality constraints and infeasible starting points.

Our approach to solving (1) employs interpolating radial basis function (RBF) models for the objective and constraint functions within a trust-region framework. The proposed algorithm is called *CONORBIT (CONstrained Optimization by Radial Basis function Interpolation in Trust-regions)* and is an extension of the ORBIT algorithm for unconstrained optimization by Wild et al. [42]. After reviewing the relevant literature in Section 2, we describe the proposed CONORBIT algorithm in Section 3. Notably, we do not include an asymptotic analysis of the algorithm, focusing instead on a more comprehensive numerical study. Section 4 describes the setup for the computational experiments used to evaluate the practical performance of CONORBIT. The problems considered include 27 test problems, a chemical process optimization problem, and an automotive application. Section 5 presents the numerical results on these problems, and comparisons with a variety of constrained optimization methods, including COBYLA [31], NOMAD [24], a sequential penalty derivative-free method called SDPEN [26], an augmented Lagrangian method with BOBYQA [34] as the subalgorithm, and ConstrLMSRBF [35]. Section 6 provides a summary and outlines future work.

2. Related work

Two main types of derivative-free algorithms for local black-box optimization are *direct search* and *model-based* methods [10]. Direct search methods typically follow geometric patterns for selecting their iterates; model-based methods maintain surrogate models of the black-box functions and use these models to select iterates. Because model-based methods are able to exploit some curvature information through the approximation models, they tend to work better than pure direct search methods on smooth functions [13]. On the other hand, direct search methods can work better when the functions are noisy or nonsmooth, and these methods are usually easy to parallelize. Hybrid approaches also have been developed in which models are used within a direct search framework; see, for example, [10]. An extensive treatment of derivative-free optimization can be found in [13].

Direct search methods for unconstrained optimization include the simplex reflection method by Nelder and Mead [28] and pattern search and its extensions [23,41]. Among the model-based methods for unconstrained optimization are trust-region methods that use quadratic interpolation models [11,32,33] and RBF interpolation models [29,42]. Examples of hybrid approaches where models are used in direct search frameworks include pattern search with Kriging [6], pattern search with RBF models [40], and pattern search guided by simplex gradients [15].

Direct search algorithms for constrained black-box optimization include mesh adaptive direct search (MADS) [3], which is an extension of pattern search that can handle black-box constraints. MADS is implemented in the NOMAD software [24], which uses various constraint-handling techniques (e.g. a progressive barrier [4]) and can employ quadratic models for the objective function [10]. Other direct search methods for constrained optimization include SDPEN (sequential penalty derivative-free algorithm) [26] and the line-search-based algorithm by Fasano et al. [18].

One of the earliest model-based methods for constrained black-box optimization is COBYLA (Constrained Optimization BY Linear Approximation, [31]), a trust-region algorithm that uses linear interpolation models for the objective and constraint functions. Brekelmans et al. [8] also developed a derivative-free trust-region approach for constrained optimization that uses the filter method by Fletcher and Leyffer [19], and builds local linear approximations of a black-box function that feeds into the objective and constraint functions. Sinoquet and Langouet [38] proposed sequential quadratic approximation, a trust-region algorithm that uses quadratic interpolation models and is an extension of NEWUOA [33] for constrained black-box optimization. In addition, Diniz-Ehrhardt et al. [16] developed derivative-free augmented Lagrangian methods that can employ model-based algorithms such as BOBYQA (Bound Optimization BY Quadratic Approximation) [34]. More recently, Augustin and Marzouk [5] developed NOWPAC, which uses p-reduced fully linear models of the objective and constraints within a trust-region framework. NOWPAC handles constraints by using an inner boundary path that guides the iterates to become strictly feasible.

3. A derivative-free trust region algorithm for constrained black-box optimization

This section describes the *CONORBIT* algorithm, which is an extension of the ORBIT algorithm [42] that specifically addresses black-box inequality constraints. CONORBIT employs a trust-region framework and uses RBFs to model the objective f and each of the constraint functions g_i. We assume that a feasible starting point is given, but the proposed method can be extended to deal with an infeasible starting point (e.g. within a multistart framework to account for multimodalities in the feasibility metric).

In the CONORBIT algorithm, bound constraints are treated as *unrelaxable* (see [25]) and hence all points generated by the algorithm satisfy the bound constraints. As a result, f and g are never evaluated at points outside of $\mathcal{B}$. Moreover, given a feasible starting point x_0, CONORBIT maintains the invariant that all iterates (i.e. the trust-region centres) remain feasible, namely $x_k \in \Omega$ for all $k \geq 0$.

3.1 *Notation and preliminaries*

Before we describe the algorithm, we first formalize our notation. As in [42, 43], we work with a general norm $\|\cdot\|_k$, where the subscript is used to indicate that the norm could change with the iteration counter k, and we let c_1 be a constant, depending only on n, so that one has an upper

bound on the 2-norm $\|\cdot\|$:

$$\|\cdot\| \leq c_1 \|\cdot\|_k, \quad \forall k. \tag{4}$$

For notational convenience, we work with interpolation sets relative to some base point $x_b \in \mathbb{R}^n$, which is typically the centre of the current trust region. Using set notation, we have $x_b + \mathcal{Y} = \{x_b + y : y \in \mathcal{Y}\}$, where $\mathcal{Y} = \{y_1, y_2, \ldots, y_{|\mathcal{Y}|}\} \subset \mathbb{R}^n$ and the function values $f(x_b + y_i)$, $i = 1, \ldots, |\mathcal{Y}|$, are known. Consequently, if m^f is a model that interpolates f at $x_b + \mathcal{Y}$, then

$$m^f(x_b + y_i) = f(x_b + y_i), \quad i = 1, \ldots, |\mathcal{Y}|. \tag{5}$$

We always interpolate f at the base point x_b; thus, we assume, without loss of generality, that $y_1 = 0 \in \mathcal{Y}$. Moreover, the RBF models that we use require at least $n+1$ interpolation points so that $|\mathcal{Y}| \geq n+1$.

By $\mathcal{R}_k(x, \Delta)$ we denote a trust region (in terms of the norm $\|\cdot\|_k$) of radius $\Delta > 0$ centred about $x \in \mathcal{B}$, $\mathcal{R}_k(x, \Delta) = \{y \in \mathbb{R}^n : \|x - y\|_k \leq \Delta\}$.

We let e denote the vector of ones and e_i the ith column of the identity matrix, the dimension of each being derived from the context. We denote the indicator (Dirac delta) function for an event a by $\mathbb{I}_{[a]}$. That is, $\mathbb{I}_{[a]} = 1$ if event a occurs, and 0 otherwise.

An important requirement for the interpolation models used in the ORBIT algorithm is that they be *fully linear* within some neighbourhood of the current iterate. This ensures that the model satisfies first-order Taylor-like error bounds within that neighbourhood. We recall some definitions and main results from [13, 42,43].

Definition 3.1 *Suppose that $x_b \in \mathbb{R}^n$ and $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable in $\mathcal{R}_k(x_b, \Delta)$. A continuously differentiable model m^f for f is said to be fully linear on $\mathcal{R}_k(x_b, \Delta)$ if there exist constants κ_{ef} and κ_{eg} such that*

$$|f(x) - m^f(x)| \leq \kappa_{ef} \Delta^2 \quad \text{and} \quad \|\nabla f(x) - \nabla m^f(x)\| \leq \kappa_{eg} \Delta, \tag{6}$$

for all $x \in \mathcal{R}_k(x_b, \Delta)$.

The following theorem from [43] provides constants κ_{ef} and κ_{eg} that satisfy the conditions in Definition 3.1. For the moment, we will work with a subset $\mathcal{Y} = \{y_1 = 0, y_2, \ldots, y_{n+1}\}$ corresponding to exactly $n+1$ interpolation points including the base point.

Theorem 3.2 *Suppose that f and m^f are continuously differentiable functions on $\mathcal{R}_k(x_b, \Delta)$ and that ∇f and ∇m^f are Lipschitz continuous on $\mathcal{R}_k(x_b, \Delta)$ with Lipschitz constants γ_f and γ_m, respectively. Further, suppose that m^f satisfies the interpolation conditions in (5) at a set of points $\{y_1 = 0, y_2, \ldots, y_{n+1}\} \subset \mathcal{R}_k(x_b, \Delta) - x_b$ such that $\|Y^{-1}\| \leq \Lambda_Y / (c_1 \Delta)$, where $Y = [y_2, \ldots, y_{n+1}] \in \mathbb{R}^{n \times n}$, for some fixed constant Λ_Y and c_1 from (4). Then, for any $x \in \mathcal{R}_k(x_b, \Delta)$,*

$$|f(x) - m^f(x)| \leq \sqrt{n} c_1^2 (\gamma_f + \gamma_m) \left(\tfrac{5}{2} \Lambda_Y + \tfrac{1}{2}\right) \Delta^2,$$
$$\|\nabla f(x) - \nabla m^f(x)\| \leq \tfrac{5}{2} \sqrt{n} \Lambda_Y c_1 (\gamma_f + \gamma_m) \Delta.$$

As mentioned in [43], Theorem 3.2 holds for general interpolation models, and the conditions on the interpolation points are equivalent to requiring that the points $\{y_1 = 0, y_2, \ldots, y_{n+1}\}$ are sufficiently affinely independent, with Λ_Y quantifying the degree of independence.

Given a set $\mathcal{D} = \{d_1, \ldots, d_{|\mathcal{D}|}\} \subset \mathcal{R}_k(x_b, \Delta) - x_b$, where $f(x_b + d_1), \ldots, f(x_b + d_{|\mathcal{D}|})$ are known, Conn et al. [12] note that a set of points $\{y_1 = 0, y_2, \ldots, y_{n+1}\} \subset \mathcal{R}_k(x_b, \Delta) - x_b$ satisfying the conditions in Theorem 3.2 can be constructed by means of LU- and QR-like algorithms. In particular, Wild et al. [42] show how to obtain such a set of interpolation points so that the

resulting RBF models are fully linear within some neighbourhood of the trust-region centre. This procedure will be described in Section 3.6.

Our last piece of notation employed in the algorithm concerns an approximate feasible region. Given a model function for the constraints $m_k^g : \mathbb{R}^n \to \mathbb{R}^q$, we define

$$\Omega_k^m = \{x \in \mathcal{B} : m_k^g(x) \leq 0\}. \tag{7}$$

We note that, without further conditions on m_k^g, it is possible that $\Omega_k^m = \emptyset$, even when the domain Ω in (3) is nonempty.

3.2 *Criticality criterion*

For our secondary termination criterion (the primary being a budget on the number of function evaluations), we will employ a projection onto a possibly nonconvex set. We first formalize what we mean by projection.

DEFINITION 3.3 *Let $\mathcal{A} \neq \emptyset$ be a closed set in $\mathbb{R}^n$. The projection set of $x \in \mathbb{R}^n$ onto $\mathcal{A}$ is*

$$\mathcal{P}_{\mathcal{A}}(x) := \{z \in \mathcal{A} \,:\, \|z - x\| \leq \|y - x\| \;\forall\, y \in \mathcal{A}\},$$

and any $z \in \mathcal{P}_{\mathcal{A}}(x)$ is called a projection of x onto $\mathcal{A}$.

We recall that $\mathcal{P}_{\mathcal{A}}(x) \neq \emptyset$ for any $x \in \mathbb{R}^n$; this follows because $\mathcal{A}$ is a nonempty, closed set and the sublevel sets of the function $\|\cdot - x\|$ are compact. Furthermore, if $\mathcal{A}$ is also convex, we can show that $\mathcal{P}_{\mathcal{A}}(x)$ is a singleton.

Next, we define the distance of a point $x \in \mathcal{A}$ to a projection set $\mathcal{P}_{\mathcal{A}}(x+u)$ for some direction u.

DEFINITION 3.4 *Given a closed set $\mathcal{A} \neq \emptyset$ in $\mathbb{R}^n$, a point $x \in \mathcal{A}$, and a nontrivial direction $u \in \mathbb{R}^n$, define*

$$d_{\mathcal{A}}(x, u) = \min\{\|x - z\| \,:\, z \in \mathcal{P}_{\mathcal{A}}(x + u)\}. \tag{8}$$

Although the projection of $x+u$ onto $\mathcal{A}$ may not be unique for nonconvex $\mathcal{A}$, by asking for the minimum distance from x to the projection set $\mathcal{P}_{\mathcal{A}}(x+u)$, the distance measure in (8) is well defined. The following proposition motivates the use of this distance as a measure of optimality.

PROPOSITION 3.5 *If $x_* \in \Omega$, Ω is convex, and f is differentiable at x_*, then $d_\Omega(x_*, -\nabla f(x_*)) = 0$ if and only if x_* satisfies the first-order necessary condition for* (1) *that*

$$\nabla f(x_*)^{\mathrm{T}}(y - x_*) \geq 0 \quad \textit{for all } y \in \Omega. \tag{9}$$

Proof If $d_\Omega(x_*, -\nabla f(x_*)) = 0$, then by the convexity of $\|x_* - \cdot\|$ and the definition in (8), $z_* = x_* \in \arg\min_{y\in\Omega} \|x_* - \nabla f(x_*) - y\|^2$. A first-order necessary condition for this problem is that $\nabla f(x_*)^{\mathrm{T}}(y - x_*) \geq 0$ for all $y \in \Omega$.

On the other hand, if $\nabla f(x_*)^{\mathrm{T}}(y - x_*) \geq 0$ for all $y \in \Omega$, then $\|x_* - \nabla f(x_*) - y\|^2 - \|x_* - \nabla f(x_*) - x_*\|^2 = \|x_* - y\|^2 - 2\nabla f(x_*)^{\mathrm{T}}(x_* - y) \geq 0$ for all $y \in \Omega$. As a consequence, $z = x_* \in \Omega$ is feasible with respect to the constraints in (8). Since $z = x_*$ attains the lower bound of zero for $\|x_* - z\|$, it follows that $d_\Omega(x_*, -\nabla f(x_*)) = 0$. ∎

3.3 *Error bound on criticality measure for convex domains*

In this section, we consider the special case when Ω_k^m is a nonempty, closed convex set and hence the projection onto Ω_k^m is unique. For such constraint sets, the following proposition holds (see, e.g. [7]).

PROPOSITION 3.6 *Let $\mathcal{A} \neq \emptyset$ be a closed convex set in $\mathbb{R}^n$. Then the projection mapping $\mathcal{P}_{\mathcal{A}} : \mathbb{R}^n \to \mathcal{A}$ is unique and satisfies $\|\mathcal{P}_{\mathcal{A}}(x) - \mathcal{P}_{\mathcal{A}}(y)\| \leq \|x - y\|, \ \forall x, y \in \mathbb{R}^n$. That is, $\mathcal{P}_{\mathcal{A}}$ is Lipschitz continuous with constant* 1.

Using this result, we can derive the following bound relating our criticality measure $d_{\Omega_k^m}(x_k, -\nabla m_k^f(x_k))$ to a criticality measure based on the objective function's gradient at x_k.

PROPOSITION 3.7 *Suppose that Ω_k^m in* (7) *is a nonempty, closed convex set, that m_k^f is fully linear on $\mathcal{R}_k(x_k, \epsilon_1)$, and that $d_{\Omega_k^m}(x_k, -\nabla m_k^f(x_k)) \leq \epsilon_2$ for some $\epsilon_1, \ \epsilon_2 > 0$. Then*

$$d_{\Omega_k^m}(x_k, -\nabla f(x_k)) \leq \kappa_{eg}\epsilon_1 + \epsilon_2,$$

where κ_{eg} is the constant from (6).

Proof Using Proposition 3.6, we have that

$$\begin{aligned} d_{\Omega_k^m}(x_k, -\nabla f(x_k)) &= \|x_k - \mathcal{P}_{\Omega_k^m}(x_k - \nabla f(x_k))\| \\ &\leq \|x_k - \mathcal{P}_{\Omega_k^m}(x_k - \nabla m_k^f(x_k))\| + \|\mathcal{P}_{\Omega_k^m}(x_k - \nabla m_k^f(x_k)) - \mathcal{P}_{\Omega_k^m}(x_k - \nabla f(x_k))\| \\ &= d_{\Omega_k^m}(x_k, -\nabla m_k^f(x_k)) + \|\mathcal{P}_{\Omega_k^m}(x_k - \nabla m_k^f(x_k)) - \mathcal{P}_{\Omega_k^m}(x_k - \nabla f(x_k))\| \\ &\leq \epsilon_2 + \|\nabla f(x_k) - \nabla m_k^f(x_k)\| \leq \kappa_{eg}\epsilon_1 + \epsilon_2, \end{aligned}$$

where the last inequality follows from Definition 3.1. ■

We note that the above bound still involves the approximation of the feasible set Ω by the set Ω_k^m. In a variety of settings, however, one can expect that $d_{\Omega_k^m}(x_k, -\nabla f(x_k)) = d_{\Omega}(x_k, -\nabla f(x_k))$, and hence this bound would also apply to the stationarity measure in Proposition 3.5. Examples where this holds include when a gradient step is feasible ($x_k - \nabla f(x_k) \in \Omega \cap \Omega_k^m$) and when the constraint functions active at $x_k - \nabla f(x_k)$ are linear and an RBF model with a linear polynomial tail is used (so that the corresponding $m_k^{g_i}$ are also linear).

3.4 *Algorithm description*

We now present Algorithm 1, which is a general derivative-free trust-region algorithm for constrained black-box optimization. The input parameters are as follows:

- η_1: Ratio threshold for the contraction and expansion of the trust region ($0 < \eta_1 < 1$)
- γ_0, γ_1: Contraction and expansion factors for the trust-region radius ($0 < \gamma_0 < 1 < \gamma_1$)
- β: Constant that provides a required minimum step-size factor before the trust region is expanded ($\beta \in (0, 1]$); since the trust-region radius also controls model quality, it may be advantageous to only increase the radius when the trust-region constraint is binding for the subproblem solution
- x_0: Feasible starting point ($x_0 \in \Omega$)
- ξ: Constraint model margin ($\xi \geq 0$)

- Δ_0: Initial trust-region radius ($\Delta_0 \in (0, \Delta_{\max}]$)
- $\Delta_{\max}$: Maximum trust-region radius ($\Delta_{\max} \in (0, \frac{1}{2}\min_i\{u_i - l_i\}]$)
- μ_f: Maximum number of (objective, constraint) function evaluations ($\mu_f \geq n + 2$)
- ε: Tolerance for criticality measure.

In Algorithm 1, the parameters are initialized in Step 0. Then, the models m_k^f and m_k^g for the objective and constraint functions are formed in Step 1.1. The interpolation set $\mathcal{Y}$ and the RBF models in this step are formed in exactly the same way as in ORBIT. The procedure for choosing $\mathcal{Y}$ is described in Algorithm 3 in Wild and Shoemaker [43] and the RBF models we used are described in Sections 3.6 and 3.7. For Algorithm 1, we require only that the models m_k^f, m_k^g interpolate f,g at $x_k + \mathcal{Y}$ and that they can be made fully linear in $\mathcal{R}_k(x_k, \Delta_k)$ in a finite number of function evaluations. We note that since $x_0 \in \Omega$ and that the update (13) moves x_k only to feasible points, it follows that $x_k \in \Omega$ for all k. As an immediate consequence of m_k^g interpolating g at x_k and $x_k \in \Omega$, it follows that Ω_k^m is always nonempty (since $x_k \in \Omega_k^m$).

Termination occurs when either the maximum number, μ_f, of evaluations has been performed or the criticality test in Step 1.2 is satisfied. We note that the distance $d_{\Omega_k^m}$ is well defined because $\Omega_k^m \neq \emptyset$.

In Step 1.3, a trust-region subproblem is solved. Using models that are twice continuously differentiable and whose derivatives are easy to calculate (which is the case for our RBF models) facilitates local solution of this subproblem using standard derivative-based methods. The trust-region subproblem involves a margin $\xi \geq 0$ on the constraint models. The purpose of this margin is to facilitate the generation of feasible iterates by requiring the solution of the trust-region subproblem to be more strict in satisfying the constraint models.

In the basic case when $\xi = 0$, we note that the subproblem (10) is simply minimizing the model of the objective function, m_k^f, subject to the trust-region constraint, bound constraints, and constraint models $m_k^g(x) \leq 0$:

$$\min\{m_k^f(x) : x \in \mathcal{R}_k(x_k, \Delta_k) \cap \Omega_k^m\}.$$

If x_k is strictly feasible in the original problem (i.e. $g(x_k) < 0$ and $x_k \in \text{int}(\mathcal{B})$), then x_k is also strictly feasible in the trust-region subproblem with $\xi = 0$. Furthermore, if m_k^g is also continuous (which is the case for the RBF models introduced next), then there exists a neighbourhood around x_k where the components of m_k^g are all strictly negative, and so the trust-region subproblem with $\xi = 0$ will have a nonempty interior.

The case when $\xi > 0$ facilitates the generation of feasible trial points by enforcing a margin for constraints that are sufficiently inactive at x_k. In the computationally expensive setting, only a limited number of objective and constraint function evaluations can be performed, and so a user may prefer methods that make it more likely to generate feasible points. The numerical experiments in Section 4 provide evidence that such a margin improves the performance of CONORBIT in the setting where (near-)feasible points are demanded. The margin is particularly helpful when one of the constraint models is active at the solution of the trust-region subproblem with $\xi = 0$. Since the model constraints are approximations of the actual constraints, a point on the boundary of a model constraint may not be on the boundary of the corresponding actual constraint. By using a margin, we are making it more likely that the iterate satisfies the actual constraints. Note that when there are many inequality constraints, only one constraint violation is sufficient to render the iterate infeasible. The idea of using a margin for constraint models was first introduced in Regis [36] and was found to be helpful for the COBRA algorithm on the MOPTA08 automotive problem [22] with 124 variables and 68 inequality constraints. The idea is somewhat similar to the inner boundary path introduced by Augustin and Marzouk [5] that guides iterates to become strictly feasible.

Algorithm 1 General constrained derivative-free algorithm.

0. Initialization. Set constants $\beta \in (0,1]$, $\eta_1 \in (0,1)$, $0 < \gamma_0 < 1 < \gamma_1 < \infty$, $\xi \geq 0$ and $\Delta_{\max} \in (0, \min_i\{u_i - l_i\}/2]$. Input $x_0 \in \Omega$, $\Delta_0 \in (0, \Delta_{\max}]$, $\varepsilon > 0$, and μ_f; set $k = 0$.

1. Check budget. While fewer than μ_f evaluations have been performed:

1.1 Form models. Obtain models m_k^f and m_k^g based on an interpolation set $x_k + \mathcal{Y}$ containing x_k.

1.2 Conduct criticality test. If $d_{\Omega_k^m}(x_k, -\nabla m_k^f(x_k)) \leq \varepsilon$:

(a) If the models m_k^f and m_k^g are not fully linear on $\mathcal{R}_k(x_k, \Delta_k)$, then make m_k^f and m_k^g fully linear on $\mathcal{R}_k(x_k, \Delta_k)$. Return to 1.

(b) ElseIf $\Delta_k > \varepsilon$, update $\Delta_k = \gamma_0 \Delta_k$, and return to 1.2.

(c) Else: exit algorithm with approximate solution x_k.

1.3 Obtain step. Approximately solve the subproblem

$$\min \left\{ m_k^f(x) : m_k^g(x) + \xi \sum_{i=1}^{q} \mathbb{I}_{[g_i(x_k) \leq -\xi]} e_i \leq 0,\ x \in \mathcal{R}_k(x_k, \Delta_k) \cap \mathcal{B} \right\}, \tag{10}$$

and let x_+ be the solution obtained.

1.4 Evaluate candidate. If $x_+ \neq x_k$, then evaluate f and g at x_+.

In any case, set

$$\rho_k = \begin{cases} -\infty & \text{if } x_+ = x_k \\ \dfrac{f(x_k) - f(x_+)}{m_k^f(x_k) - m_k^f(x_+)} & \text{else.} \end{cases} \tag{11}$$

1.5 Update trust region. Adjust the trust region according to the ratio ρ_k, the feasibility of x_+, and the quality of the models m_k^f and m_k^g:

$$\Delta_{k+1} = \begin{cases} \min\{\gamma_1 \Delta_k, \Delta_{\max}\} & \text{if } x_+ \in \Omega \text{ and } \rho_k \geq \eta_1 \text{ and } \|x_+ - x_k\|_k \geq \beta \Delta_k, \\ \gamma_0 \Delta_k & \text{if } x_+ \in \Omega \text{ and } \rho_k < \eta_1 \text{ and} \\ & \quad \text{the models } m_k^f, m_k^g \text{ are fully linear on } \mathcal{R}_k(x_k, \Delta_k), \\ \gamma_0 \Delta_k & \text{if } x_+ \notin \Omega \text{ and} \\ & \quad \text{the models } m_k^f, m_k^g \text{ are fully linear on } \mathcal{R}_k(x_k, \Delta_k), \\ \Delta_k & \text{otherwise.} \end{cases} \tag{12}$$

$$x_{k+1} = \begin{cases} x_+ & \text{if } x_+ \in \Omega \text{ and } \rho_k \geq \eta_1, \\ x_+ & \text{if } x_+ \in \Omega \text{ and } 0 < \rho_k < \eta_1 \text{ and} \\ & \quad \text{the models } m_k^f, m_k^g \text{ are fully linear on } \mathcal{R}_k(x_k, \Delta_k), \\ x_k & \text{otherwise.} \end{cases} \tag{13}$$

1.6 Model improvement. If $x_{k+1} = x_k$, m_k^f and m_k^g are not fully linear on $\mathcal{R}_k(x_k, \Delta_k)$, and x_+ is not a model-improving point:

Evaluate f and g at a model-improving point $x_- \in \mathcal{R}_k(x_k, \Delta_k) \cap \mathcal{B}$ (obtained from Algorithm 2).

1.7 Iterate. Update $k \leftarrow k + 1$, and return to 1.

The form of subproblem (10) tightens each nonlinear constraint model that is sufficiently inactive (as measured by the user-input $\xi \geq 0$). The indicator $\mathbb{I}_{[g_i(x_k) \leq -\xi]}$ is employed to ensure that the feasible region of (10) remains nonempty for all values of ξ. In practice, it may be beneficial to use different margins, $\xi_1, \ldots, \xi_q$, for different constraint functions and/or to update the margin width ξ as the algorithm progresses. For simplicity, here we keep the scalar value ξ fixed for all the constraint functions and all iterations.

As described above, Algorithm 1 maintains feasible iterates x_k. However, the solution of the trust-region subproblem x_+ might coincide with the current centre of the trust region x_k. This is the reason for our extended-real-valued ρ_k update in Step 1.4. In addition to preventing a repeated evaluation of the expensive f,g at $x_+ = x_k$, the update ensures that a model-improving point x_- will be the next point evaluated. Such evaluation either will be based on the current trust-region radius Δ_k (when the models are not fully linear on $\mathcal{R}_k(x_k, \Delta_k)$) or based on a radius $\gamma_0^j \Delta_k$, occurring after j consecutive radius reductions have caused the models to be no longer fully linear on the updated trust region $\mathcal{R}_k(x_{k+j}, \Delta_{k+j}) = \mathcal{R}_k(x_k, \gamma_0^j \Delta_k)$.

In Step 1.5, the trust region is updated according to the feasibility of the trial point x_+, the ratio ρ_k of the actual improvement to the predicted improvement, and the quality (as measured by the model being fully linear on the current trust region) of the models for the objective and constraint functions. In particular, the trust-region centre is moved to x_+ if x_+ is feasible and either ρ_k is at least the threshold η_1 or $0 < \rho_k < \eta_1$ and the models for the objective and constraints are fully linear within the current trust region. In all other cases, the trust-region centre remains the same. The trust-region radius is possibly increased if x_+ is feasible, the actual reduction was significant ($\rho_k \geq \eta_1$), and the step size was sufficiently close to being bound by the trust region ($\|x_+ - x_k\|_k \geq \beta \Delta_k$). Moreover, the trust-region radius is decreased if the models for the objective and constraints are fully linear within the current trust region and either x_+ is feasible and $\rho_k < \eta_1$ or x_+ is infeasible. In all other cases, the trust-region radius remains the same.

In Step 1.6 we consider the case when the trust-region remains unchanged and thus the quality of the local models should be improved. We avoid doing an evaluation when the subproblem solution x_+ is deemed to improve the quality of the models. When x_+ does not improve the models, we follow a procedure based on that in [42] but modified for the constrained setting as described in Section 3.5.

3.5 *Model-improving points*

We now describe our revision of model-improving points for the generally constrained case.

Given an interpolation set $\mathcal{Y}$ containing 0, a trust region $\mathcal{R}_k(x_k, \Delta_k)$, and positive constants $c \geq 1$ and $\theta_1 \in (0, 1]$, the ORBIT algorithm certifies models m_k as being fully linear on $\mathcal{R}_k(x_k, \Delta_k)$ if there are n points, $y_1, \ldots, y_n$, in $\mathcal{Y}$ with the following properties:

M1. $y_i \in \mathcal{R}_k(0, c\Delta_k)$, $i = 1, \ldots, n$;
M2. $x_k + y_i \in \mathcal{B}$, $i = 1, \ldots, n$; and
M3. the pivots of $QR = (1/\Delta_k)[y_1 \cdots y_n]$ satisfy $|r_{ii}| \geq \theta_1$, $i = 1, \ldots, n$.

We recall from Section 3.1 and Theorem 3.2 that these properties allow us to establish that the resulting models are fully linear in $\mathcal{R}_k(x_k, \Delta_k)$.

Provided that $x_k \in \mathcal{B}$, one can always find such points by choosing appropriate signs for the coordinate directions $\Delta_k e_1, \ldots, \Delta_k e_n$. This follows because, for every $i = 1, \ldots, n$, at least one of $x_k \pm \Delta_k e_i$ is contained in $\mathcal{B}$ since $\Delta_k \leq \Delta_{\max} \leq \frac{1}{2}(u_i - l_i)$ for all k. The existence of these coordinate directions ensures that the RBF models can be made fully linear in any trust region by performing at most n additional evaluations of f and g. Such additional evaluations, however, are wasteful in practice and can be avoided by exploiting the affine independence of nearby,

previously evaluated points. Hence, ORBIT produces an orthonormal basis $z_1, \ldots, z_p \subset \mathbb{R}^n$ (of size $p \leq n - \text{rank}(\mathcal{Y})$) orthogonal to directions in $\mathcal{Y}$ determined to be sufficiently independent (as measured by the parameter θ_1). ORBIT then determines a model-improving point $y_- = \alpha_1 z_1 + \cdots + \alpha_p z_p$ that satisfies the conditions M1–M3 when added to these sufficiently independent directions.

Formally, the procedure shown in Algorithm 2 is performed.

Algorithm 2 General bound-constrained model-improvement procedure.

0. Initialization. Given orthonormal directions $z_1, \ldots, z_p$, x_k, Δ_k, and parameter θ_1.

1. Loop over directions. For $i = 1, \ldots, p$:

2.1 Check projection Let $t_+ = \max_{t \leq 1}\{t : x_k + t\Delta_k z_i \in \mathcal{B}\}$ and $t_- = \max_{t \leq 1}\{t : x_k - t\Delta_k z_i \in \mathcal{B}\}$.

- If $t_+ \geq \theta_1$: Set $x_- = x_k + t_+\Delta_k z_i$ and `return`.
- Elseif $t_- \geq \theta_1$: Set $x_- = x_k - t_-\Delta_k z_i$ and `return`.
- Elseif $i = p$: `Return`, reset candidate model-improving directions to $e_1, \ldots, e_n$.
- Else continue to next i.

The directions input by ORBIT to Algorithm 2 are orthogonal to a partial set $y_1, \ldots, y_j$ (with $j < n$) that satisfies M1 and M2, and whose incomplete ($j < n$) pivots satisfy M3. The procedure then ensures that the output direction $x_- - x_k$ satisfies these same conditions, with an additional pivot satisfying M3. When no output is provided, the algorithm indicates that the directions need to be rotated, an action that may result in the evaluation of up to n model-improving points (associated with the coordinate directions, which are guaranteed to result in termination of Algorithm 2 as discussed above regarding the parameter θ_1).

Since the interpolation sets used in CONORBIT are common to both the objective and constraints models, this procedure can also be used in CONORBIT to obtain fully linear models. Furthermore, in implementations of ORBIT, we choose the signs and the order of the directions $z_1, \ldots, z_p$ based on predicted objective values $m_k^f(x_k \pm \Delta_k z_i)$. Since the procedure ignores the general constraints g, however, the resulting model-improving point $x_- = x_k + y_- \in \mathcal{R}_k(x_k, \Delta_k) \cap \mathcal{B}$ may be infeasible (with respect to g) and thus will be of little benefit to future iterations.

Consequently, in CONORBIT we modify the procedure to favour points that are more likely, as predicted by the current models, to be feasible. Currently this modification is done by determining the sign and ordering of the directions $z_1, \ldots, z_p$ to minimize the number of predicted constraint violations, $\sum_{j=1}^{q} \mathbb{I}_{[m_k^{g_j}(x_k \pm \Delta_k z_i) > 0]}$.

Unfortunately, because the t_+ (or t_-) values produced by Algorithm 2 can be much less than1, this strategy alone may not be effective. Ideally, one wants to solve the equivalent of a trust-region subproblem in the subspace spanned by $z_1, \ldots, z_p$. When the margin is enforced, this would look like

$$\min_{\alpha} \left\{ m_k^f(x_k + \Delta_k Z\alpha) : m_k^g(x_k + \Delta_k Z\alpha) + \xi \sum_{i=1}^{q} \mathbb{I}_{[g_i(x_k) \leq -\xi]} e_i \leq 0, \right.$$
$$\left. x_k + \Delta_k Z\alpha \in \mathcal{R}_k(x_k, \Delta_k) \cap \mathcal{B} \right\}. \tag{14}$$

If the approximate solution $\alpha \in \mathbb{R}^p$ obtained satisfies $\|\alpha\| \geq \theta_1$, then the model-improving point $x_- = x_k + \Delta_k Z\alpha$ satisfies M1–M3.

If the obtained $\|\alpha\| < \theta_1$, then we solve the nonconvex problem

$$\min_{\alpha}\{\|\max\{0, m_k^g(x_k + \Delta_k Z\alpha)\}\| : \|\alpha\| \geq \theta_1,\ x_k + \Delta_k Z\alpha \in \mathcal{R}_k(x_k, \Delta_k) \cap \mathcal{B}\}, \tag{15}$$

which attempts to minimize the constraint violation, while keeping α sufficiently bounded away from zero (through θ_1). We can again output $x_- = x_k + \Delta_k Z\alpha$ since it satisfies M1–M3. If no solution to (15) is obtained (e.g. because the feasible region $\{\alpha : \|\alpha\| \geq \theta_1,\ x_k + \Delta_k Z\alpha \in \mathcal{R}_k(x_k, \Delta_k) \cap \mathcal{B}\}$ is empty), then we again resort to coordinate directions in place of Z.

3.6 *RBF models*

We now briefly review RBFs, which we use to model both the objectives and nonlinear constraints in CONORBIT.

For convenience, we take $g_0 := f$ so that the objective and constraint functions are collectively denoted by $g_0, g_1, \ldots, g_q$. Given a current iterate x_k and a single set of interpolation points $\mathcal{Y} = \{y_j\}_{j=1}^{|\mathcal{Y}|}$, where $g_i(x_k + y_j)$ is known for $i = 0, 1, \ldots, q$, $j = 1, \ldots, |\mathcal{Y}|$, we fit RBF models $m_k^f = m_k^{g_0}$ and $m_k^g = (m_k^{g_1}, m_k^{g_2}, \ldots, m_k^{g_q})$ of the form

$$m_k^{g_i}(x_k + s) = \sum_{j=1}^{|\mathcal{Y}|} \lambda_j^{g_i} \phi(\|s - y_j\|) + p^{g_i}(s) \quad \text{for } i = 0, 1, \ldots, q.$$

Here, $\phi : \mathbb{R}_+ \to \mathbb{R}$ is a radial function that could take one of several forms as discussed next, $\lambda_j^{g_i} \in \mathbb{R}$, and p^{g_i} is a polynomial.

We now state a property of RBFs that we will employ when forming our models.

Definition 3.8 *Let $\pi = [\pi_1, \ldots, \pi_{\hat{p}}]$ be a basis for n-variate polynomials of degree $d-1$, with the convention that $\pi = \emptyset$ if $d = 0$. A function $\phi : \mathbb{R}_+ \to \mathbb{R}$ is said to be conditionally positive definite of order $d \geq 0$ if for all sets of distinct points $\mathcal{Y} \subset \mathbb{R}^n$ and all $\lambda \neq 0$ satisfying $\sum_{j=1}^{|\mathcal{Y}|} \lambda_j \pi(y_j) = 0$, the quadratic form $\sum_{i=1}^{|\mathcal{Y}|} \sum_{j=1}^{|\mathcal{Y}|} \lambda_i \lambda_j \phi(\|y_i - y_j\|)$ is positive.*

Wild and Shoemaker [43] provide a list of popular twice continuously differentiable RBFs and their order of conditional positive definiteness. For example, the cubic form (with radial function $\phi(r) = r^3$) is conditionally positive definite of order 2, whereas the Gaussian form (with radial function $\phi(r) = \exp(-r^2/\gamma^2)$, where $\gamma > 0$ is a parameter) is conditionally positive definite of order 0. Note that if ϕ is conditionally positive definite of order d, then it is also conditionally positive definite of order $\hat{d} \geq d$.

3.7 *Forming RBF models*

We now provide the details for forming the RBF models that are used in CONORBIT. Given a radial function, polynomial order and basis of dimension $\hat{p}$, and an interpolation set $\mathcal{Y}$ of $|\mathcal{Y}| \geq \hat{p}$ distinct points, we define $\Phi \in \mathbb{R}^{|\mathcal{Y}| \times |\mathcal{Y}|}$ by $\Phi_{i,j} = \phi(\|y_i - y_j\|)$ and $\Pi \in \mathbb{R}^{\hat{p} \times |\mathcal{Y}|}$ by $\Pi_{i,j} = \pi_i(y_j)$ for $i = 1, \ldots, \hat{p}$ and $j = 1, \ldots, \|\mathcal{Y}\|$. RBF models are then determined by solving the following linear system with multiple right-hand sides that is an extension of the interpolation equation in Powell [30]:

$$\begin{bmatrix} \Phi & \Pi^{\mathrm{T}} \\ \Pi & 0_{\hat{p} \times \hat{p}} \end{bmatrix} \begin{bmatrix} \lambda^{g_0} & \lambda^{g_1} & \ldots & \lambda^{g_q} \\ \nu^{g_0} & \nu^{g_1} & \ldots & \nu^{g_q} \end{bmatrix} = \begin{bmatrix} G_0 & G_1 & \ldots & G_q \\ 0_{\hat{p} \times 1} & 0_{\hat{p} \times 1} & \ldots & 0_{\hat{p} \times 1} \end{bmatrix}, \tag{16}$$

where $G_i = [g_i(x_k + y_1), \ldots, g_i(x_k + y_{|\mathcal{Y}|})]^{\mathrm{T}}$ for $i = 0, 1, \ldots, q$.

The coefficient matrix of (16) is symmetric but it is generally indefinite. However, the system can be solved in a numerically stable manner by relying on the conditional positive definiteness of ϕ and by requiring that $\hat{p}$ of the points in $\mathcal{Y}$ be poised for interpolation by a $d-1$ degree polynomial. This latter condition is equivalent to requiring that Π has full row rank, a property that is maintained for the interpolation sets used in both ORBIT and CONORBIT. Now consider the truncated QR factorization $\Pi^{\mathrm{T}} = QR$. Since Π^{T} has full column rank, R is nonsingular. From (16), we have

$$\Pi\lambda^{g_i} = 0_{\hat{p}\times 1} \quad \text{for } i = 0, 1, \ldots, q.$$

Let $Z \in \mathbb{R}^{(|\mathcal{Y}|-\hat{p})\times(|\mathcal{Y}|-\hat{p})}$ be the orthogonal matrix whose columns form an orthonormal basis for Null(Π). Then for all $i = 0, 1, \ldots, q$, $\lambda^{g_i} = Z\omega^{g_i}$ for some $\omega^{g_i} \in \mathbb{R}^{|\mathcal{Y}|-\hat{p}}$ and (16) reduces to

$$Z^{\mathrm{T}}\Phi Z w^{g_i} = Z^{\mathrm{T}} G_i, \quad R v^{g_i} = Q^{\mathrm{T}}(G_i - \Phi Z\omega^{g_i}), \quad i = 0, 1, \ldots, q. \tag{17}$$

It follows from ϕ being conditionally positive definite of order d that $Z^{\mathrm{T}}\Phi Z$ is positive definite, and hence these solutions are well defined.

In CONORBIT, we will strive for (but do not require on all iterations) fully linear RBF models. Consequentially, we will require that the radial function ϕ used be conditionally positive definite of order $d = 2$ (this includes all the radial functions mentioned in [43]). Note that in this case we will also require a linear polynomial tail, so that $\hat{p} = n + 1$ and we require that $\mathcal{Y}$ contain $n+1$ points that are affinely independent.

Because a common interpolation set is employed for $m^f, m^{g_1}, \ldots, m^{g_p}$, certifying that any one of these RBF models is fully linear is equivalent to certifying that all these models are fully linear (provided that $f, g_1, \ldots, g_p$ all satisfy the regularity conditions assumed in Theorem 3.2).

The formation of the RBF models in Step 1.1 of Algorithm 1 now identically follows that in ORBIT but with $q+1$ right-hand sides. In particular, we use the same additional input parameters for forming the RBF models from [42, 43]: $p_{\max} \geq n+1$ (maximum number of interpolation points) and θ_i for $i = 1, \ldots, 4$ satisfying $\theta_2 > 0$, $\theta_4 \geq \theta_3 \geq 1$ and $0 < \theta_1 \leq 1/\theta_3$. Details about these parameters are found in Algorithm 3 and Section 4.3 of [43].

3.8 *Demonstration of CONORBIT on a two-dimensional problem*

To illustrate the CONORBIT (2-norm) algorithm in two dimensions, we consider the following constrained optimization problem (modified from [20] to have a single nonlinear constraint for ease of illustration and also to have a quadratic objective):

$$\begin{aligned}\min\{&(x-0.2)^2 + (y+0.1)^2 : 1.5 - x - 2y - 0.5\sin(2\pi(x^2 - 2y)) \leq 0,\\ &0 \leq x \leq 1, 0 \leq y \leq 1\}.\end{aligned} \tag{18}$$

Table 1. Parameter settings for CONORBIT (2-norm and ∞-norm).

Parameter	Value	Parameter	Value
Δ_0	0.2	β	0.5
$\Delta_{\max}$	0.5	ε	10^{-8}
$p_{\max}$	$2n+1$	ξ	10^{-6}
η_0	0	θ_1	10^{-3}
η_1	0.2	θ_2	10^{-7}
γ_0	1/2	θ_3	2
γ_1	2	θ_4	$\sqrt{n}$

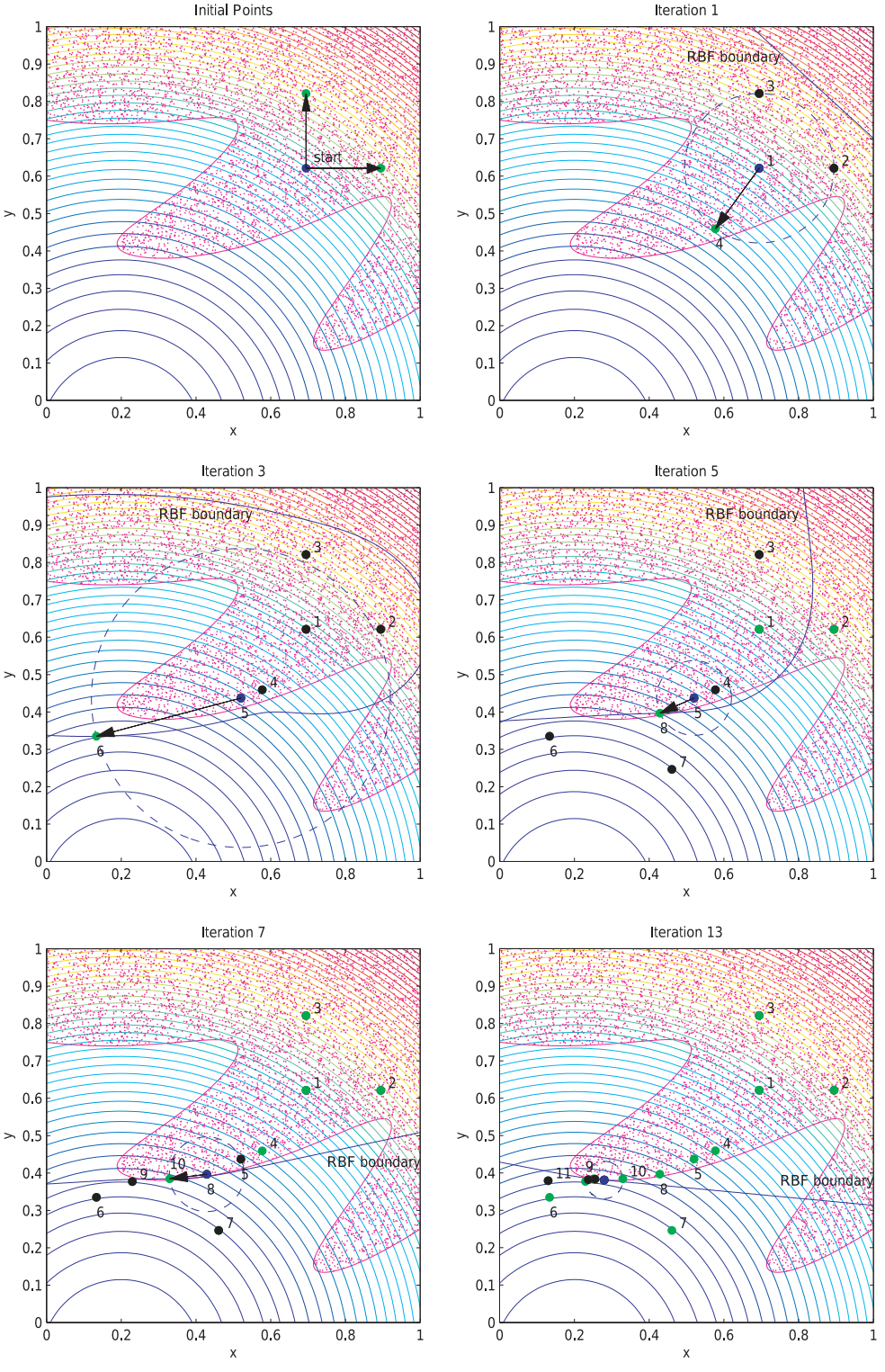

Figure 1. Initial function evaluations and iterations 1, 3, 5, 7, and 13 of CONORBIT (2-norm) in (18).

We employ a cubic RBF model ($\phi(r) = r^3$) with a linear polynomial tail, do not use a margin ($\xi = 0$), and use $\|\cdot\|_k = \|\cdot\|$ for all k. This demo uses the default parameter settings for CONORBIT (2-norm) reported in Table 1 in Section 4.4 except for the value of the constraint model margin.

Figure 1 (top left) shows the feasible region of the problem and the contour lines of the objective function. Note that the feasible region (covering the northeast region) is nonlinear and nonconvex and that the direction of improvement for the objective function is southwest. The starting point is at $x_0 = [0.7, 0.6]$. In order to build the initial RBF models, the objective and constraint functions are also evaluated at the points $[0.9, 0.6]$ and $[0.7, 0.8]$, which are $\Delta_0 = 0.2$ units away from the starting point along the positive coordinate directions. Next, Figure 1 (top right) shows the boundary of the RBF constraint (i.e. $m_k^g(x) = 0$), the initial trust region and the current step as determined from the trust-region subproblem (10). We note that the RBF models are all linear since there are only $n+1$ points. We also note that the RBF constraint model is not active at the subproblem solution.

The rest of the plots in Figure 1 (middle left) to (bottom right) show the trajectory of CONORBIT during iterations 3, 5, 7, and 13, respectively. Observe that the RBF constraint boundaries are now nonlinear and that CONORBIT has generated points that were infeasible with respect to the nonlinear constraint (points 6 and 7). However, the trust-region centre always remains feasible. Moreover, observe that the RBF model constraint is generally a crude global approximation of the actual constraint. In later iterations, the RBF constraint model provides a good local approximation of the actual constraint within the trust region.

The numerical experiments below suggest that the RBF models of the constraints need not be accurate global approximations of the actual constraints in order to be useful. What is critical is that the models be able to provide a good local approximation of the constraint functions and the feasible and infeasible regions *within the trust region*.

4. Numerical experiment setup

In this section we summarize the problems and algorithms considered in our numerical experiments, the results of which are described in Section 5. All computations are performed in Matlab 7.11.0 using an Intel Core i7 CPU 860 2.8 GHz desktop machine.

4.1 *MOPTA08: automotive design optimization*

The first problem we consider is a multidisciplinary optimization problem from the auto industry derived from the problem posed by Jones [22] during the MOPTA 2008 conference. The goal is to determine the values of decision variables (e.g. shape variables) that minimize the mass of an automotive vehicle subject to performance constraints (e.g. crashworthiness, durability). Because our proposed method is designed for relatively low-dimensional constrained black-box optimization problems, the actual problem we use is a smaller version of the original 124-dimensional MOPTA08 problem [22]: only 12 decision variables are allowed to vary, while the rest of the decision variables are fixed to the corresponding values for the best solution found by Regis [36]. As in the original MOPTA08 problem, there are 68 black-box inequality constraints and $[0, 1]$ bound constraints. This smaller problem is referred to as MOPTA08-12D.

The original MOPTA08 problem is available as a Fortran code in [1], and the settings for the decision variables that have been fixed to obtain the MOPTA08-12D problem are provided in the Appendix. Matlab interfaces for MOPTA08-12D and the original MOPTA08 problem along with the initial feasible points used in this study are available at http://people.sju.edu/~rregis/pages/software.html.

The MOPTA08-12D problem has smooth objective and constraint functions because these functions are based on Kriging response surfaces to a proprietary automotive design problem. The simulation time needed to evaluate the objective and constraints takes about 0.32 sec, while each simulation of the real version could take many minutes.

4.2 *Styrene: chemical process optimization*

We also consider a chemical process optimization problem called the *Styrene* problem. The goal is to maximize the net production value (NPV) of the Styrene production process. The process of Styrene production underlying this problem is discussed in [39], and the simulation-based optimization of this process was studied in [2].

In theory, the objective NPV is a smooth function that depends on the operating costs, sales, income tax and actualization rates, depreciation, and investment. These components depend on process variables such as reactor dimensions, heater and cooler temperatures, pump pressure, and air fraction. In addition, there exist process-based (e.g. purity levels and environmental regulations) and economic-based constraints on the variables. The dependency of NPV and the constraints on the variables generally cannot be expressed in closed form.

We work with the same black-box simulator used in [2]. There are $n = 8$ decision variables and $q = 11$ black-box inequality constraints. The black-box simulator has hidden constraints, in the sense that there are domains $\Omega^f \subset \mathcal{B}$ and $\Omega^g \subset \mathcal{B}$ such that the simulator returns $f(x) = M$ for $x \notin \Omega^f$ and $g(x) = Me$ for $x \notin \Omega^g$, where M is a large value (10^{20}) and e is the vector of 1's in $\mathbb{R}^{11}$. In our tests of the simulator, we have observed that $\Omega^f = \{x \in \mathcal{B} : g_8(x) < \bar{c}\}$, where $\bar{c} > 0$ (i.e. the function value M is returned if g_8 goes above a threshold). We note that $\Omega^f \subset \Omega^g$. The first four constraints are binary (taking values in $\{0, 1\}$), which in [25] is referred to as *nonquantifiable* constraints. We note that because of these characteristics, the Styrene problem clearly violates the smoothness assumption (C2). For feasible points, the simulator takes around one-and-a-half seconds to evaluate the function and constraints.

4.3 *Test problems*

To facilitate tests on a broader set of problems, we also consider 27 test problems from the literature. The set comprises the problems used by Conn and Le Digabel [10] that have inequality constraints, 20 problems used by Regis [35,36] that are widely used in the engineering optimization literature, and four engineering design problems: WB4 (welded beam design), GTCD4 (gas transmission compressor design), PVD4 (pressure vessel design), and SR7 (speed reducer design). The number of variables and constraints in these problems are given in Table A1 in the Appendix.

Like the MOPTA08-12D and Styrene problems, the objective and constraint functions of these problems are not expensive to evaluate. However, it is still possible to do meaningful comparisons between CONORBIT and the alternative methods by pretending that the test problems are computationally expensive as was done in previous studies [35,36]. For example, if each simulation of the Crescent10 problem takes one hour, then the results presented can still provide an idea of relative performance of the algorithms after some fixed computational budget (e.g. after 100 simulations, which takes roughly 100 hours). We expect that the relative performance of the algorithms on the test problems will mimic their performance on truly expensive problems whose surfaces resemble those of the test problems. In addition, as was done in previous studies [35,36] , the objective or constraint functions of some of the test problems are rescaled either by dividing by some positive constant or by applying a logarithmic transformation without changing the feasible region and the location of the optima before any algorithms are applied.

The purpose of this rescaling is to make the function values less extreme and avoid problems with fitting RBF models. In particular, the plog transformation from [37] was applied to some of the objective and constraint functions:

$$\text{plog}(x) = \begin{cases} \log(1+x) & \text{if } x \geq 0, \\ -\log(1-x) & \text{if } x < 0, \end{cases}$$

where log is the natural logarithm. This transformation is differentiable, strictly increasing, symmetric with respect to the origin, and it tones down extremely high or extremely negative function values. More details of the rescaled test problems are found in [36].

4.4 *Alternative methods and parameter settings*

Two implementations of CONORBIT (2-norm and ∞-norm versions) are compared with the following alternative methods: SDPEN [26], NOMAD [3,24], COBYLA [31], AUGLAG [9,16] with BOBYQA [34] as the subalgorithm, and the ConstrLMSRBF heuristic [35]. NOMAD is available through the OPTI toolbox [14]. COBYLA, AUGLAG, and BOBYQA are all available through the NLopt package [21], which is also available through the OPTI toolbox.

NOMAD is an implementation of MADS [3] and SDPEN is a sequential penalty derivative-free algorithm. COBYLA is a derivative-free trust-region method that uses linear interpolation models for the objective and constraint functions. AUGLAG is an implementation of an augmented Lagrangian algorithm, and BOBYQA is a derivative-free trust-region algorithm for bound-constrained optimization that uses a minimum Frobenius norm quadratic model. In addition, ConstrLMSRBF is a heuristic method that uses RBF models of the objective and constraint functions to select function evaluation points from a set of randomly generated candidate points. In every iteration, ConstrLMSRBF chooses its function evaluation point to be the best candidate point according to two criteria—predicted objective function value and minimum distance from previously evaluated points—from among the candidate points with the minimum number of predicted constraint violations.

The values of the CONORBIT parameters we used are summarized in Table 1. Note that these parameters are set with the assumption that the region defined by the bound constraints has been scaled to the unit cube $[0,1]^n$. A cubic RBF model with a linear polynomial tail is used for both the tested CONORBIT algorithms (2-norm and ∞-norm versions). Moreover, in addition to the given starting point, the CONORBIT algorithms use n additional initial points obtained by moving Δ_0 units from the starting point in each of the n positive coordinate directions. If $x_0 + \Delta_0 e_i$ violates one of the bound constraints, then this point is replaced by $x_0 - \Delta_0 e_i$, and this will satisfy the bound constraints provided $\Delta_0 \leq 0.5$.

As mentioned earlier, Regis [36] found that a small margin on the RBF constraints helped improve the performance of an algorithm that uses RBF models for the constraints. Hence, we performed extensive tests to check whether such a margin is also helpful for CONORBIT. That is, we ran CONORBIT with various settings for the margin $\xi \geq 0$ so that Step 1.3 of Algorithm 1 approximately solves the trust-region subproblem (10). If (10) turns out to be infeasible, then the margin is removed (i.e. $\xi = 0$) only on the constraints where the margin causes this subproblem to become infeasible. The purpose of the margin is to make the solution to the trust-region subproblem strictly feasible with respect to as many RBF inequality constraints as possible, and thereby increase the likelihood that the next iterate will be a feasible point. The results are discussed in Section 5.4. These experiments suggest that a value of $\xi = 10^{-6}$ works well on the problems in this study, and hence this will be the default value of ξ when comparing CONORBIT with the other methods.

In addition to a margin on the RBF constraint models, we also introduce a *delay trust-region reduction heuristic* where the trust-region radius is *not* reduced even when the iterates are infeasible and the models are fully linear. We found that implementing this heuristic for some fixed number h of simulations substantially improves the short-term performance of CONORBIT since it prevents the trust-region from rapidly shrinking in the early stages of running the algorithm. Numerical results for this heuristic are discussed in Section 5.5. Hence, for the comparison with other methods, this heuristic is implemented in CONORBIT for $h = 10(n + 1)$ simulations where the iterates are infeasible and the models are fully linear.

We use the NOMAD software [24] and the NLopt software [21] through the OPTI toolbox [14] that runs in Matlab. The NLopt software includes implementations of COBYLA and an augmented Lagrangian method called AUGLAG. We run AUGLAG using BOBYQA as the subalgorithm. These methods that are run through the OPTI toolbox will be referred to as OPTI-NOMAD, OPTI-COBYLA, and OPTI-AUGLAG-BOBYQA. We also developed a Matlab implementation of SDPEN, which we call SDPENm. Default values for OPTI-NOMAD, OPTI-COBYLA, OPTI-AUGLAG-BOBYQA, and SDPENm are used in all comparisons. For ConstrLMSRBF, a Latin hypercube design of size $n+1$ is used for initialization, and the type of RBF model used is the same one used by the CONORBIT algorithms. Moreover, the parameter values for ConstrLMSRBF are the same as those used in [35].

In all numerical comparisons in this study, each algorithm is run 30 times on each of the 27 test problems and 10 times on the Styrene and MOPTA08-12D applications. Each run corresponds to a different feasible starting point, but the same feasible starting point is used for the different algorithms in a given run. Moreover, the stochastic algorithm ConstrLMSRBF is only run once for each starting point on a given test problem, and so, it is also run a total of 30 times for each test problem. Since this study focuses only on the case where a feasible starting point is provided, the numerical work to obtain the feasible starting points for the different problems is performed in the background and is not included in the comparisons. To get an idea of the difficulty of finding feasible starting points for the test problems, the reader is referred to Regis [36], where an RBF heuristic was developed to handle problems where no feasible starting points are provided. In addition, for some of the test problems, approximate values of the ratio of the size of the feasible region to the size of the search space are available in the literature.

4.5 *Limitation of the comparative experiments*

Before we compare CONORBIT with other methods in the next section, we recall that we used default parameter settings for the alternative methods. However, it might still be possible to improve their performance on the test problems used by tuning their parameters. Our goal in this paper is *not* to prove that the proposed CONORBIT algorithm is superior to the other methods. Rather, we simply wish to make the case that CONORBIT ought to be considered as a promising method for constrained black-box optimization, by demonstrating that it performs reasonably well on a wide variety of test problems and on two important applications.

5. Results and discussion

We compare the CONORBIT variants with alternatives using performance and data profiles [17,27]. We take a problem p to be defined by the pairing of a test problem (from Table A1) and a starting point. Since we have 27 test problems and 30 starting points, we consider $27 \times 30 = 810$ problems in total. Given a set $\mathcal{P}$ of problems, we run a set $\mathcal{S}$ of solvers. In the comparisons, there are six solvers: CONORBIT (2-norm), OPTI-NOMAD, OPTI-COBYLA, OPTI-AUGLAG-BOBYQA, SDPENm, and ConstrLMSRBF. Although CONORBIT (∞-norm)

was also run, we do not show the results to avoid clutter in the plots since they are similar to those for the 2-norm version.

For any pair $(p, s) \in \mathcal{P} \times \mathcal{S}$, the *performance ratio* is

$$r_{p,s} = \frac{t_{p,s}}{\min\{t_{p,s} : s \in \mathcal{S}\}},$$

where the performance measure $t_{p,s}$ is the number of simulations required to satisfy the convergence test described below. Here, one simulation yields the objective function value $f(x)$ and constraint function values $g(x)$ for a given input x. Clearly, $r_{p,s} \geq 1$ for any $p \in \mathcal{P}$ and $s \in \mathcal{S}$. Moreover, for a given problem p, the best solver s for this problem attains $r_{p,s} = 1$. Furthermore, by convention, $r_{p,s} = \infty$ whenever solver s fails to yield a solution that satisfies the convergence test.

For any solver $s \in \mathcal{S}$ and for any $\alpha \geq 1$,

$$\rho_s(\alpha) = \frac{1}{|\mathcal{P}|}|\{p \in \mathcal{P} : r_{p,s} \leq \alpha\}|$$

gives the fraction of problems where the performance ratio is at most α. The performance profile $\rho_s(\alpha)$ measures the performance of solver s relative to the solvers in $\mathcal{S}$ on the problem set $\mathcal{P}$.

Next, given a solver $s \in \mathcal{S}$ and $\alpha > 0$, the *data profile* of a solver s on problem set $\mathcal{P}$ is

$$d_s(\alpha) = \frac{1}{|\mathcal{P}|}\left|\left\{p \in \mathcal{P} : \frac{t_{p,s}}{n_p + 1} \leq \alpha\right\}\right|,$$

where $t_{p,s}$ is again the number of simulations required by solver s to satisfy the convergence test on problem p and n_p is the number of variables in problem p. For a given solver s and any $\alpha > 0$, $d_s(\alpha)$ is the fraction of problems 'solved' (i.e. problems where the solver generated a point satisfying the convergence test) by s within $\alpha \cdot (n_p + 1)$ simulations (equivalent to α simplex gradient estimates) [27].

In the context of derivative-free, expensive constrained black-box optimization, algorithms are compared given a fixed and relatively limited number of simulations. Hence, our convergence test uses tolerances $\epsilon, \tau > 0$, and it checks whether a point x obtained by a solver satisfies

$$\max_{i=1,\ldots,q} g_i(x) \leq \epsilon, \quad \text{and} \quad f(x^{(0)}) - f(x) \geq (1-\tau)(f(x^{(0)}) - f_L^{\epsilon,\mathcal{S},\mu_f}). \tag{19}$$

Here, $f_L^{\epsilon,\mathcal{S},\mu_f}$ is the minimum objective function value of ϵ-feasible points obtained by *any* of the solvers within a given budget μ_f of simulations. That is, if x^{ϵ,s,μ_f} is the best ϵ-feasible point obtained by solver $s \in \mathcal{S}$ within μ_f simulations, then $f_L^{\epsilon,\mathcal{S},\mu_f} = \min_{s\in\mathcal{S}} f(x^{\epsilon,s,\mu_f})$.

Since the starting point $x^{(0)}$ is feasible in all our tests, it follows that at least one solver $s \in \mathcal{S}$ will satisfy the convergence test (19) for any given $\epsilon, \tau, \mu_f > 0$. In cases where there are multiple points from solver s satisfying (19) on problem p, the performance measure $t_{p,s}$ is taken to be the minimum number of evaluations needed to satisfy (19).

5.1 *Comparison with alternative methods on test problems*

Figure 2 shows the data profiles of the various solvers on all 27 test problems in this study. Here, the tolerances for the convergence test are $\epsilon = 10^{-8}$ and $\tau = 0.01$. These data profiles are calculated up to a maximum number of simulations that is equivalent to 150 simplex gradient estimates. Recall that one simulation yields the values of $f(x)$ and $g(x)$ for a given input x.

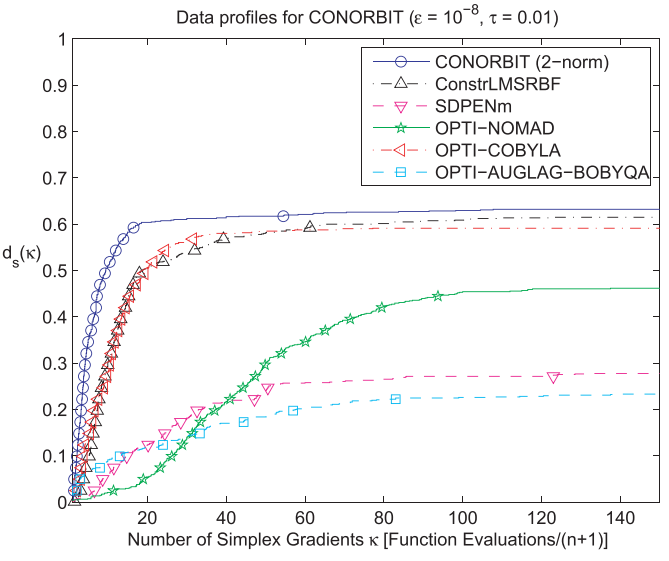

Figure 2. Data profiles of the algorithms on the test problems with 30 different feasible starting points, $\epsilon = 10^{-8}$ and $\tau = 0.01$ in (19). The test problems have 2–20 decision variables and 1–38 inequality constraints.

Figure 3 shows the performance profiles of the solvers given a computational budget of 200 simulations and 300 simulations on all 27 test problems.

The data profiles in Figure 2 show that CONORBIT (2-norm with $\xi = 10^{-6}$ and with the delay trust-region reduction heuristic) is generally better than the alternatives on the test problems. As mentioned earlier, the delay trust-region reduction heuristic prevents the trust-region from rapidly shrinking when the iterates are infeasible and the models are fully linear within the trust-region during the early stages of running the algorithm. In particular, CONORBIT is much better than SDPENm, OPTI-AUGLAG-BOBYQA, and OPTI-NOMAD, as can be seen from the wide gap between the corresponding data profiles. Moreover, CONORBIT is an improvement over OPTI-COBYLA and ConstrLMSRBF. For example, the data profiles in Figure 2 show that CONORBIT (2-norm) satisfies the convergence test in about 60% of the problems within 100 simplex gradient estimates. In contrast, SDPENm and OPTI-AUGLAG-BOBYQA satisfy the convergence test for less than 30% of the problems, and OPTI-NOMAD satisfies the convergence test for less than 50% of the problems within the same computational budget.

Next, the performance profiles after 200 or 300 simulations in Figure 3 show that the fraction of problems for which CONORBIT is the best is about 30%, while the corresponding fractions for all alternative methods (other than ConstrLMSRBF) are around 10% or less. Furthermore, the fraction of problems where the performance ratio is at most $\alpha = 4$ after 200 or 300 simulations is more than 60% for CONORBIT, around 50% for OPTI-COBYLA and ConstrLMSRBF, and less than 15% for the other methods.

We also plotted the corresponding data and performance profiles where the tolerance ϵ was set to 0. The profiles for all the solvers except OPTI-COBYLA did not really change. For OPTI-COBYLA, we note a substantial deterioration in performance, indicating that this algorithm was using an internal default constraint tolerance that is $\leq 10^{-8}$. To be fair to OPTI-COBYLA, we set $\epsilon = 10^{-8}$ in all profiles.

In addition, since COBYLA is the only model-based trust region method that directly handles constraints among the alternative methods, we identified the problems for which CONORBIT (2-norm) performed better than OPTI-COBYLA. In particular, we looked at the data profiles for all the solvers on each of the 27 test problems and compared CONORBIT (2-norm) with

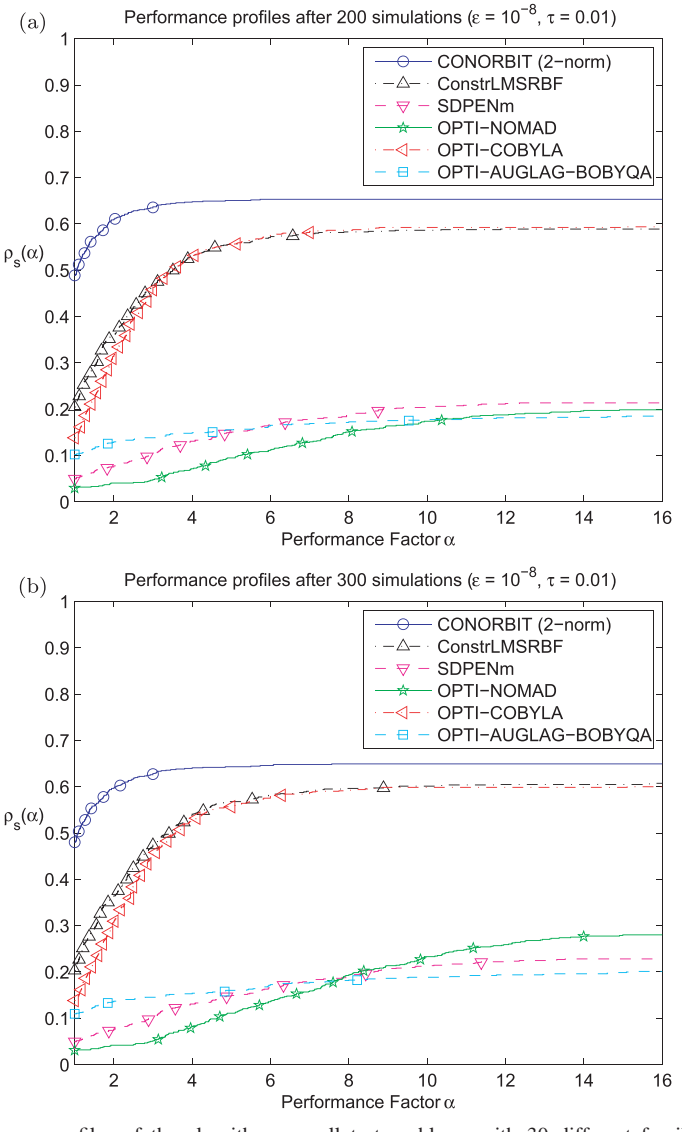

Figure 3. Performance profiles of the algorithms on all test problems with 30 different feasible starting points, $\epsilon = 10^{-8}$ and $\tau = 0.01$ in (19). The test problems have 2–20 decision variables and 1–38 inequality constraints. (a) After 200 simulations. (b) After 300 simulations.

OPTI-COBYLA. The individual data profiles show that CONORBIT (2-norm) is better or at least competitive with OPTI-COBYLA on 22 of the 27 test problems (all except G1, G13MOD, G18, WB4, and Crescent10), most of which have nonlinear objective functions and constraint functions. Among these five test problems where OPTI-COBYLA performed better are G1, which is a linear programming problem, and Crescent10, which has a very simple linear objective and has quadratic constraints. Two other problems where OPTI-COBYLA performed better are WB4 and G13MOD, which only involve 4 and 5 decision variables, respectively.

These results provide further evidence that the use of interpolation models for the black-box constraints can yield substantial performance improvements in the computationally expensive setting. Neither OPTI-NOMAD nor SDPENm uses models for the constraints, while OPTI-COBYLA uses linear interpolation models for the black-box objective and constraint functions,

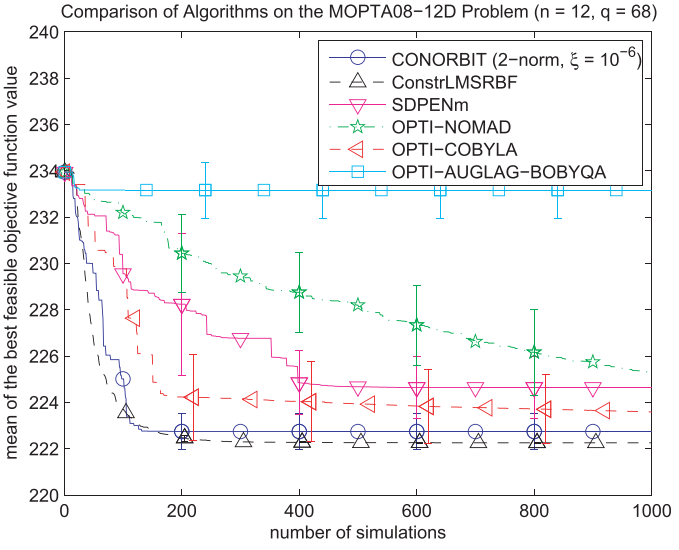

Figure 4. Mean of the best feasible objective function value versus the number of simulations for the MOPTA08 problem. The error bars for the stochastic algorithms correspond to 95% *t* confidence intervals for the mean.

and ConstrLMSRBF uses the same type of RBF model for the black-box functions as CONORBIT. Moreover, although OPTI-AUGLAG-BOBYQA uses quadratic interpolation models of the augmented Lagrangian function, this does not appear to be as effective as building models for the constraints. The performance of SDPENm, OPTI-NOMAD, and OPTI-AUGLAG-BOBYQA will most likely improve if they also use models for the constraints.

5.2 *Comparison with alternative methods on the automotive application*

Figure 4 shows the graphs of the mean of the best feasible objective function value obtained by various algorithms versus the number of simulations on the MOPTA08-12D problem. We refer to these graphs as *average progress curves*. Recall that each algorithm is run for 10 trials on the MOPTA08-12D problem, where each trial begins with a different feasible starting point. The error bars correspond to 95% *t* confidence intervals for the mean.

The average progress curves show much better performance for CONORBIT compared with OPTI-NOMAD, OPTI-AUGLAG-BOBYQA, and SDPENm up to 1000 simulations on the MOPTA08-12D problem. Moreover, CONORBIT (2-norm) shows better performance than OPTI-COBYLA. In addition, the average performance of CONORBIT (2-norm) is close to that of ConstrLMSRBF.

The MOPTA08-12D problem is smooth, so model-based methods such as CONORBIT, COBYLA, and ConstrLMSRBF are expected to perform well. However, the results for OPTI-COBYLA suggest that RBF models are better than linear interpolation models in approximating the 68 black-box constraints in the MOPTA08-12D problem and effectively guiding the trust-region steps. Moreover, the results for OPTI-AUGLAG-BOBYQA suggest that using interpolation models for the augmented Lagrangian function is not as effective as using interpolation models directly on the black-box constraints.

5.3 *Performance of CONORBIT on the styrene problem*

Since CONORBIT is a model-based method, it is not expected to perform well on problems where the surfaces of the objective or constraint functions are highly nonsmooth and also on

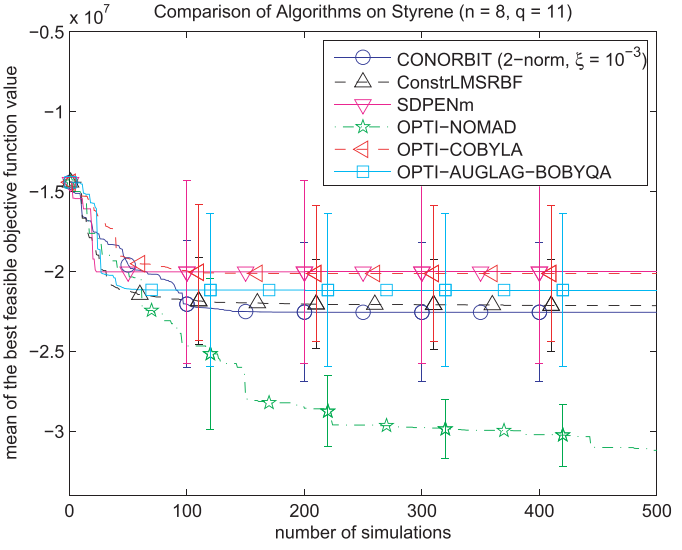

Figure 5. Mean of the best feasible objective function value versus the number of simulations for the Styrene problem. The error bars for the stochastic algorithms correspond to 95% t confidence intervals for the mean.

problems with hidden constraints or binary constraints such as the Styrene problem. With a hidden constraint, the simulator typically returns a large value for the objective and constraint functions when it is violated. Moreover, a binary constraint function is highly discontinuous. Hence, the presence of hidden or binary constraints generally results in poor approximation of these constraints using interpolation models.

Figure 5 shows the average progress curves for CONORBIT and the alternative methods on the Styrene problem with a computational budget of 500 simulations. As with the MOPTA08 problem, each algorithm is run for 10 trials, where each trial begins with a different feasible starting point.

As mentioned earlier, this problem is highly nonsmooth because of the presence of binary constraint values and because of the handling of its hidden constraints. Recall that if a point violates the hidden constraints, the Styrene code yields a high value of 10^{20} for the objective and all constraint functions. Here, CONORBIT handles the Styrene problem the same way as the other problems. That is, the objective and constraint values of 10^{20} and also the binary constraint values are used for RBF interpolation just like the other data points. The average progress curves on the Styrene problem show that OPTI-NOMAD is much better than the other methods, followed by the CONORBIT and ConstrLMSRBF. It might be possible to modify CONORBIT to work better on the Styrene problem but this is beyond the scope of the paper and will be considered in future work.

5.4 *Effect of the RBF constraint margin on the performance of CONORBIT*

Extensive testing was performed to study the effect of having a small margin on the RBF models of the inequality constraints in CONORBIT. Figure 6(a) shows the data profiles for CONORBIT (2-norm) with different values for the RBF constraint margin: $\xi = 0,\ 10^{-6},\ 10^{-4},\ 10^{-3}$, and 10^{-2}. Recall that the bound constraints for all problems were rescaled to $[0, 1]^n$ so these values for ξ are based on this rescaling. Figure 6(b) shows the performance profiles for CONORBIT with various RBF constraint margins after 300 simulations, respectively. Similar data and performance profiles were obtained for CONORBIT (∞-norm). These profiles are calculated on all 27 test

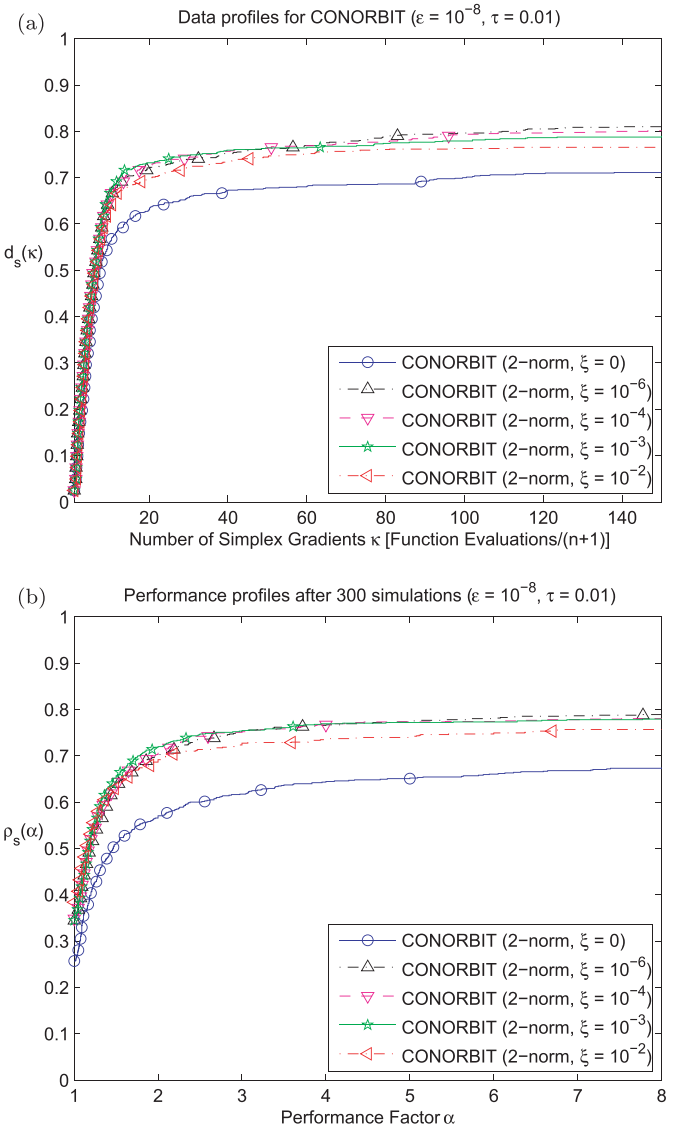

Figure 6. Data and performance profiles for CONORBIT (2-norm) with various RBF constraint margins on all test problems with 30 different feasible starting points, $\epsilon = 10^{-8}$ and $\tau = 0.01$. (a) Data profiles. (b) Performance profiles after 300 simulations.

problems with 30 different feasible starting points. Hence, each profile plot has $27 \times 30 = 810$ problems and five solvers. As before, the tolerances for the convergence test are $\epsilon = 10^{-8}$ and $\tau = 0.01$.

The data and performance profiles consistently show that using an RBF constraint margin improves the performance of CONORBIT. Moreover, these profiles show that the performance of CONORBIT is only slightly sensitive to the choice of the margin ξ. In particular, CONORBIT (2-norm or ∞-norm) with an RBF constraint margin of anywhere from 10^{-6} up to 10^{-2} performs much better than CONORBIT without any RBF constraint margin. A possible reason for this robust behaviour with respect to the margin is that the margin of an RBF constraint is removed whenever it causes the trust region subproblem to become infeasible. However, the profiles also

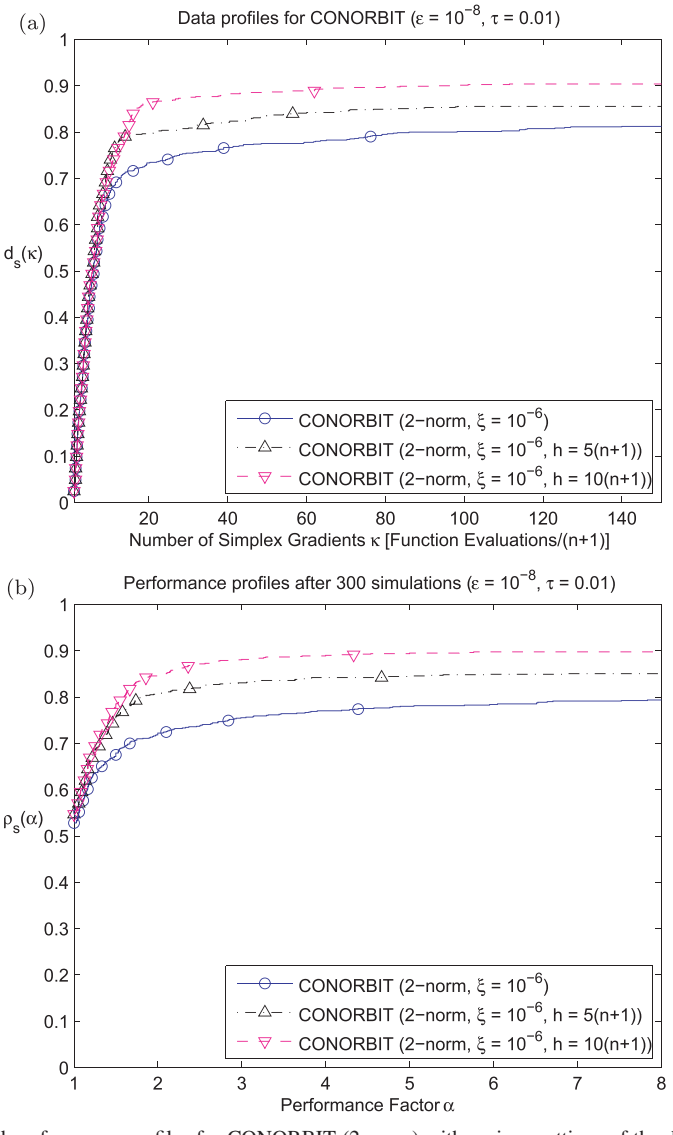

Figure 7. Data and performance profiles for CONORBIT (2-norm) with various settings of the delay heuristic on all test problems with 30 different feasible starting points, $\epsilon = 10^{-8}$ and $\tau = 0.01$. (a) Data profiles. (b) Performance profiles after 300 simulations.

show that when the margin is too large (e.g. when $\xi = 10^{-2}$), the performance of CONORBIT deteriorates slightly. Figure 6 consistently show that an RBF constraint margin of either 10^{-4} or 10^{-6} appear to be good choices for the test problems used. In all other experiments as well as in the default parameter settings for CONORBIT, we use a margin of $\xi = 10^{-6}$.

5.5 *Improving the practical performance of CONORBIT*

The previous section showed that the performance of CONORBIT can be improved by using a small margin on the RBF constraints. This section shows that the performance of CONORBIT can be further improved by delaying the shrinking of the trust region when the iterate turns out to be infeasible. Recall that the trust region is reduced when the iterate is infeasible and the models

are fully linear within the trust region. In the numerical experiments, the iterates end up infeasible in many iterations even when the models are fully linear, and this results in the rapid shrinking of the trust regions. This result is expected because, for highly constrained problems, generating feasible iterates early in the search is not likely since the models might not be accurate enough even if they are fully linear within the trust region. As a result of the smaller trust regions, the progress of CONORBIT was relatively slow on many of the problems.

To deal with this issue, we introduced a *delay trust-region reduction heuristic* in CONORBIT where the trust region is *not* reduced for some fixed number of simulations, denoted by h, even when the iterate is infeasible and the models are fully linear within the trust region. Note that this heuristic can also be interpreted as an initialization procedure that is implemented before the actual trust-region iterations are performed. To test this idea, we ran CONORBIT (2-norm and ∞-norm) with a margin of $\xi = 10^{-6}$ on all the test problems (30 trials per problem as before) where this delay heuristic is set at $h=0$, $h = 5(n + 1)$ and $h = 10(n + 1)$. The data profiles for CONORBIT (2-norm) are shown in Figure 7(a), and the performance profiles after 300 simulations are shown in Figure 7(b). Similar results were obtained for CONORBIT (∞-norm) with the same settings for this heuristic. The results show an advantage of implementing this delay heuristic in both the 2-norm and ∞-norm versions of CONORBIT. In particular, a setting of $h = 10(n + 1)$ yields the best result among the values of h considered. Hence, the default value of the delay heuristic will be set at $h = 10(n + 1)$ along with a margin of $\xi = 10^{-6}$ when CONORBIT is compared with alternative methods.

5.6 *Running times on the automotive application problem*

Table 2 shows the overhead running times (i.e. excluding total times spent on function evaluations) of the CONORBIT algorithms and alternatives on the MOPTA08-12D problem after 1000 simulations. For example, the overhead running time of CONORBIT (2-norm) on MOPTA08-12D is 788.29 sec (13.14 minutes) for 1000 simulations.

Clearly, the running times of the CONORBIT algorithms on MOPTA08-12D are much longer than those of the non-model-based alternative methods such as OPTI-NOMAD and SDPEN. For truly expensive functions, the overhead running times for CONORBIT are still much smaller than the total time spent on function evaluations. For example, if each simulation of the MOPTA08-12D problem takes 1 hour, then the mean running time of CONORBIT (2-norm) for 1000 simulations would be 1000.22 hours, while the mean running time of SDPENm for 1000 simulations would be 1000.0003 hours. However, from Figure 4, the mean of the best feasible objective value obtained by CONORBIT (2-norm) is better than that obtained by SDPENm after 1000 simulations.

Table 2. Average running times (over 10 trials) of the different algorithms on 1000 simulations of the MOPTA08-12D problem with 12 decision variables and 68 black-box inequality constraints (excluding time spent on simulations) on an Intel Core i7 CPU 860 2.8 GHz desktop machine.

Algorithm	Average running time (sec)
CONORBIT (2-norm)	788.29
CONORBIT (∞-norm)	829.36
SDPENm	1.16
OPTI-NOMAD	66.05
OPTI-COBYLA	2.38
OPTI-AUGLAG-BOBYQA	2.38
ConstrLMSRBF	156.48

We did not report the average overhead running times on the test problems because most of these are smaller than the overhead running times on MOPTA08-12D. Note that if the simulation time for one of the test problems happens to be expensive, then the total running time of an algorithm is dominated by the total time spent on simulations.

6. Summary

We developed the CONORBIT algorithm, which is an extension of the ORBIT algorithm that can be used for problems with black-box inequality constraints. CONORBIT is a trust-region algorithm that uses interpolating RBF models satisfying Definition 3.8, for the objective and each of the constraint functions. Moreover, CONORBIT is a feasible point algorithm, and this is important when dealing with constrained, expensive black-box problems where only a relatively limited number of function evaluations can be performed. The 2-norm and ∞-norm implementations of CONORBIT were compared with COBYLA, NOMAD, SDPEN, an augmented Lagrangian algorithm with BOBYQA as the subalgorithm, and ConstrLMSRBF on 27 test problems, the Styrene chemical process optimization problem, and an automotive problem with 12 decision variables and 68 black-box inequality constraints. Performance and data profiles show that the CONORBIT algorithms performed better than SDPEN, NOMAD, COBYLA, ConstrLMSRBF, and the augmented Lagrangian method on the test problems. Moreover, average progress curves show that the CONORBIT algorithms performed better than the other alternatives and are competitive with ConstrLMSRBF on the automotive application. However, the CONORBIT algorithms did not perform as well as NOMAD on the nonsmooth Styrene problem with hidden constraints. The results on Styrene demonstrate the limitations of CONORBIT and possibly other model-based methods on nonsmooth problems with hidden constraints. In addition, this study found that using a small margin on the RBF constraints and delaying the reduction of the trust region when the iterates are infeasible are both helpful in improving the practical performance of CONORBIT on the test problems. Overall, CONORBIT is a promising algorithm for constrained, expensive black-box optimization.

Acknowledgments

The authors thank Don Jones from General Motors Product Development for providing a Fortran simulation program for the MOPTA08 problem. The U.S. Government retains for itself, and others acting on its behalf, a paid-up, nonexclusive, irrevocable worldwide licence in said article to reproduce, prepare derivative works, distribute copies to the public, and perform publicly and display publicly, by or on behalf of the Government.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This material was based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research [under contract DE-AC02-06CH11357]. The first author was supported by a 2012 summer research grant from Saint Joseph's University.

References

[1] M.F. Anjos, MOPTA 2008 benchmark website, 2009. Available at: http://www.miguelanjos.com/jones-benchmark.
[2] C. Audet, V. Béchard and S. Le Digabel, *Nonsmooth optimization through mesh adaptive direct search and variable neighborhood search*, J. Global Optim. 41 (2008), pp. 299–318.

[3] C. Audet and J.E. Dennis, Jr. *Mesh adaptive direct search algorithms for constrained optimization*, SIAM J. Optim. 17 (2006), pp. 188–217.
[4] C. Audet and J.E. Dennis, Jr. *A progressive barrier for derivative-free nonlinear programming*, SIAM J. Optim. 20 (2009), pp. 445–472.
[5] F. Augustin and Y.M. Marzouk, *NOWPAC: A provably convergent nonlinear optimizer with path-augmented constraints for noisy regimes*, Tech. Rep. 1403.1931, 2014, arXiv.
[6] A.J. Booker, J.E. Dennis, P.D. Frank, D.B. Serafini, V. Torczon and M.W. Trosset, *A rigorous framework for optimization of expensive functions by surrogates*, Struct. Optim. 17 (1999), pp. 1–13.
[7] J.M. Borwein and A.S. Lewis, *Convex Analysis and Nonlinear Optimization: Theory and Applications*, 2nd ed., CMS Books in Mathematics, Springer, New York, 2006.
[8] R. Brekelmans, L. Driessen, H. Hamers and den Hertog D., *Constrained optimization involving expensive function evaluations: A sequential approach*, Eur. J. Oper. Res. 160 (2005), pp. 121–138.
[9] A.R. Conn, N.I.M. Gould and Toint P., *A globally convergent augmented lagrangian algorithm for optimization with general constraints and simple bounds*, SIAM J. Numer. Anal. 28 (1991), pp. 545–572.
[10] A.R. Conn and S. Le Digabel, *Use of quadratic models with mesh-adaptive direct search for constrained black box optimization*, Optim. Methods Softw. 28 (2013), pp. 139–158.
[11] A.R. Conn, K. Scheinberg and P. Toint, *Recent progress in unconstrained nonlinear optimization without derivatives*, Math. Program. 79 (1997), pp. 397–414.
[12] A.R. Conn, K. Scheinberg and L.N. Vicente, *Geometry of interpolation sets in derivative free optimization*, Math. Program. 111 (2008), pp. 141–172.
[13] A.R. Conn, K. Scheinberg and L.N. Vicente, *Introduction to Derivative-Free Optimization*, MPS/SIAM Series on Optimization, Society for Industrial and Applied Mathematics, Philadelphia, PA, 2009.
[14] J. Currie and D.I. Wilson, *OPTI: Lowering the Barrier between Open Source Optimizers and the Industrial MATLAB User*, in *Foundations of Computer-Aided Process Operations*, 8–11 January, FOCAPO, Savannah, GA, 2012.
[15] A.L. Custódio and L.N. Vicente, *Using sampling and simplex derivatives in pattern search methods*, SIAM J. Optim. 18 (2007), pp. 537–555.
[16] M.A. Diniz-Ehrhardt, J.M. Martínez and L.G. Pedroso, *Derivative-free methods for nonlinear programming with general lower-level constraints*, Comput. Appl. Math. 30 (2011), pp. 19–52.
[17] E.D. Dolan and J.J. Moré, *Benchmarking optimization software with performance profiles*, Math. Program. 91 (2002), pp. 201–213.
[18] G. Fasano, G. Liuzzi, S. Lucidi and F. Rinaldi, *A linesearch-based derivative-free approach for nonsmooth constrained optimization*, SIAM J. Optim. 24 (2014), pp. 959–992.
[19] R. Fletcher and S. Leyffer, *Nonlinear programming without a penalty function*, Math. Program. 91 (2002), pp. 239–269.
[20] R.B. Gramacy, G.A. Gray, S. Le Digabel, H.K.H. Lee, P. Ranjan, G. Wells and S.M. Wild, *Modeling an augmented Lagrangian for blackbox constrained optimization*, Technometrics 58 (2016), pp. 1–11.
[21] S.G. Johnson, The NLopt nonlinear-optimization package. Available at http://ab-initio.mit.edu/nlopt.
[22] D.R. Jones, *Large-scale multi-disciplinary mass optimization in the auto industry*, in *MOPTA 2008, Modeling, Optimization: Theory and Applications Conference*, August, MOPTA, Ontario, Canada, 2008.
[23] T.G. Kolda, R.M. Lewis and V. Torczon, *Optimization by direct search: New perspectives on some classical and modern methods*, SIAM Review 45 (2003), pp. 385–482.
[24] S. Le Digabel, *Algorithm 909: NOMAD: Nonlinear optimization with the MADS algorithm*, ACM Trans. Math. Softw. 37 (2011), pp. 44:1–44:15.
[25] S. Le Digabel and S.M. Wild, *A taxonomy of constraints for simulation-based optimization*, preprint (2015), ANL/MCS-P5350-0515, Argonne National Laboratory, Mathematics and Computer Science Division. Available at http://www.mcs.anl.gov/papers/P5350-0515.pdf.
[26] G. Liuzzi, S. Lucidi and M. Sciandrone, *Sequential penalty derivative-free methods for nonlinear constrained optimization*, SIAM J. Optim. 20 (2010), pp. 2614–2635.
[27] J.J. Moré and S.M. Wild, *Benchmarking derivative-free optimization algorithms*, SIAM J. Optim. 20 (2009), pp. 172–191.
[28] J.A. Nelder and R. Mead, *A simplex method for function minimization*, Comput. J. 7 (1965), pp. 308–313.
[29] R. Oeuvray and M. Bierlaire, *BOOSTERS: A derivative-free algorithm based on radial basis functions*, Int. J. Modell. Simul. 29 (2009), pp. 26–36.
[30] M.J.D. Powell, *The theory of radial basis function approximation in 1990*, in *Advances in Numerical Analysis, Volume 2: Wavelets, Subdivision Algorithms and Radial Basis Functions*, W. Light, ed., Oxford University Press, Oxford, 1992, pp. 105–210.
[31] M.J.D. Powell, *A direct search optimization methods that models the objective and constraint functions by linear interpolation*, in *Advances in Optimization and Numerical Analysis*, S. Gomez and J. Hennart, eds., Kluwer, Dordrecht, 1994, pp. 51–67.
[32] M.J.D. Powell, *UOBYQA: Unconstrained optimization by quadratic approximation*, Math. Program. 92 (2002), pp. 555–582.
[33] M.J.D. Powell, *The NEWUOA software for unconstrained optimization without derivatives*, in *Large-Scale Nonlinear Optimization*, G.D. Pillo and M. Roma, eds., Springer, 2006, pp. 255–297. Available at http://www.springer.com/us/book/9780387300634.

[34] M.J.D. Powell, *The BOBYQA algorithm for bound constrained optimization without derivatives*, Tech. Rep. DAMTP 2009/NA06, Department of Applied Mathematics and Theoretical Physics, University of Cambridge, Cambridge, 2009.
[35] R.G. Regis, *Stochastic radial basis function algorithms for large-scale optimization involving expensive black-box objective and constraint functions*, Comput. Operat. Res. 38 (2011), pp. 837–853.
[36] R.G. Regis, *Constrained optimization by radial basis function interpolation for high-dimensional expensive black-box problems with infeasible initial points*, Eng. Optim. 46 (2014), pp. 218–243.
[37] R.G. Regis and C.A. Shoemaker, *A quasi-multistart framework for global optimization of expensive functions using response surface models*, J. Global Optim. 56 (2013), pp. 1719–1753.
[38] D. Sinoquet and H. Langoüet, *SQA: A generic trust region derivative free optimization method for black box industrial applications*, in *ICCOPT International Conference on Continuous Optimization*, MOS, Lisbon, Portugal, 2013.
[39] J.D. Snyder and B. Subramaniam, *A novel reverse flow strategy for ethylbenzene dehydrogenation in a packed-bed reactor*, Chem. Eng. Sci. 49 (1994), pp. 5585–5601.
[40] Le Thi H.A., A.I.F. Vaz and L.N. Vicente, *Optimizing radial basis functions by D.C. programming and its use in direct search for global derivative-free optimization*, TOP 20 (2012), pp. 190–214.
[41] V. Torczon, *On the convergence of pattern search algorithms*, SIAM J. Optim. 7 (1997), pp. 1–25.
[42] S.M. Wild, R.G. Regis and C.A. Shoemaker, *ORBIT: Optimization by radial basis function interpolation in trust-regions*, SIAM J. Sci. Comput. 30 (2008), pp. 3197–3219.
[43] S.M. Wild and Shoemaker C., *Global convergence of radial basis function trust region derivative-free algorithms*, SIAM J. Optim. 21 (2011), pp. 761–781.

Appendix. Information on test problems and automotive application

Table A1 summarizes the test problems used in our numerical experiments.

Table A1. Constrained optimization test problems: n is the number of decision variables and q is the number of inequality constraints.

Test problem	n	q	Smooth	Global minimum or best known value
Crescent10	10	2	Yes	-9.0
Disk10	10	1	Yes	$-10\sqrt{3}$
HS114S	10	11	Yes	-2864.101066
HS114NS	9	4	No	-4628.621326
MAD6	5	7	No	0.101831
Pentagon	6	15	No	-1.859617
Snake	2	2	Yes	0.08098
SR7	7	11	Yes	2994.42
WB4	4	6	Yes	1.7250
GTCD4	4	1	Yes	2964893.60
PVD4	4	3	Yes	5804.45
G2	10	2	No	-0.67
G3MOD	20	1	Yes	-0.693152
G4	5	6	Yes	-30665.539
G5MOD	4	5	Yes	5126.50
G6	2	2	Yes	-6961.8139
G7	10	8	Yes	24.3057
G8	2	2	Yes	-0.0958
G9	7	4	Yes	680.6301
G10	8	6	Yes	7049.3307
Hesse	6	6	Yes	-310
G1	13	9	Yes	-15
G13MOD	5	3	Yes	0.0035
G16	5	38	Yes	-1.9052
G18	9	13	Yes	-0.8660
G19	15	5	Yes	32.6556
G24	2	2	Yes	-5.5080
MOPTA08-12D	12	68	Yes	222.2324
Styrene	8	11	No	-33539100

The algorithms in this paper were tested on a 12-dimensional version of the 124-dimensional MOPTA08 problem from Jones [22] where only 12 decision variables are allowed to vary while the rest of the decision variables are fixed to the corresponding values for the best solution found in [36]. The indices of the variables whose values are 0 in the best solution for MOPTA08 in [36] are as follows:

$$\{2, 3, 4, 5, 11, 12, 13, 14, 15, 17, 18, 19, 20, 21, 23, 24, 25, 28, 29, 33, 45, 46, 51, 52, 54, 55,$$

$$57, 58, 62, 65, 69, 72, 73, 75, 77, 80, 85, 90, 95, 96, 98, 101, 109, 114, 121, 123, 124\}.$$

The indices of the variables whose values are 1 in the best solution for MOPTA08 in [36] are $\{10, 34, 35, 102, 110, 112, 113, 115\}$. The settings of the other variables in the best solution for MOPTA08 found in [36] are given in Table A2.

To create the MOPTA08-12D problem, the indices of the variables that are allowed to vary in the original MOPTA08 problem are $\{2, 42, 45, 46, 49, 51, 60, 66, 72, 78, 82, 95\}$. The variables corresponding to these indices are allowed to vary between 0 and 1 as in the original problem.

Table A2. Variable settings in the original MOPTA08 problem to create the MOPTA08-12D problem.

Variable index	Value	Variable index	Value
1	0.424901536845442	70	0.204560994583241
6	0.070368163337397	71	0.221383426733140
7	0.191758133431455	74	0.094398339150782
8	0.659581587650616	76	0.308945500999398
9	0.312511963893922	78	0.320954495292004
16	0.500523655520237	79	0.735573944494521
22	0.007323697294187	81	0.510228205500305
26	0.691424942460969	82	0.558204538496431
27	0.549284506769106	83	0.527207739609047
30	0.634896580136233	84	0.827802688106254
31	0.246440520422731	86	0.665687979989158
32	0.312017956345506	87	0.724380930097933
36	0.317952882489170	88	0.287131241910361
37	0.855170249994022	89	0.414606427865539
38	0.658962493142454	91	0.419521821597494
39	0.567339401149320	92	0.388002834344305
40	0.191865333482370	93	0.029815463457816
41	0.906569943802102	94	0.631619810505481
42	0.612762717451940	97	0.192333290055701
43	0.033974026079705	99	0.692543616440458
44	0.250858210690466	100	0.669383104588378
47	0.397548021112299	103	0.729141260482755
48	0.050453268214836	104	0.161236677500071
49	0.144718492777280	105	0.561446507760961
50	0.065661350680142	106	0.487479618244102
53	0.401813779445987	107	0.728572274406474
56	0.181619715253390	108	0.240635584133437
59	0.114195768540203	111	0.383816619144691
60	0.393419805218654	116	0.152841300847909
61	0.047400618876390	117	0.158379914361422
63	0.877814656956643	118	0.210185493857930
64	0.767294038418826	119	0.889704350522390
66	0.853786893241821	120	0.791266802460270
67	0.603638877543724	122	0.931445345989215
68	0.330387862928772		