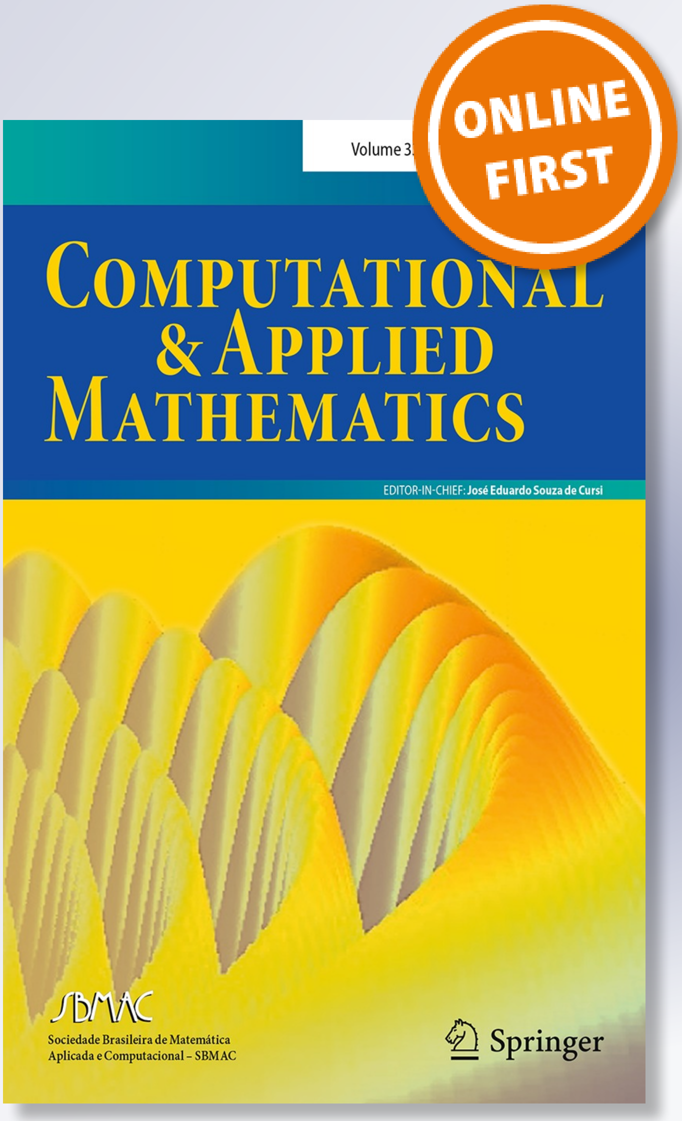
A derivative-free trust-region algorithm for composite nonsmooth optimization

Geovani Nunes Grapiglia, Jinyun Yuan & Ya-xiang Yuan

Computational and Applied Mathematics

ISSN 0101-8205

Comp. Appl. Math.
DOI 10.1007/s40314-014-0201-4


Volume 3
ONLINE FIRST
COMPUTATIONAL & APPLIED MATHEMATICS
EDITOR-IN-CHIEF: José Eduardo Souza de Cursi
SBMAC
Sociedade Brasileira de Matemática Aplicada e Computacional – SBMAC
Springer

Comp. Appl. Math.
DOI 10.1007/s40314-014-0201-4

A derivative-free trust-region algorithm for composite nonsmooth optimization

Geovani Nunes Grapiglia · Jinyun Yuan · Ya-xiang Yuan

Received: 24 September 2014 / Revised: 27 October 2014 / Accepted: 28 October 2014

Abstract The derivative-free trust-region algorithm proposed by Conn et al. (SIAM J Optim 20:387–415, 2009) is adapted to the problem of minimizing a composite function $\Phi(x) = f(x) + h(c(x))$, where f and c are smooth, and h is convex but may be nonsmooth. Under certain conditions, global convergence and a function-evaluation complexity bound are proved. The complexity result is specialized to the case when the derivative-free algorithm is applied to solve equality-constrained problems. Preliminary numerical results with minimax problems are also reported.

Keywords Nonsmooth optimization · Nonlinear programming · Trust-region methods · Derivative-free optimization · Global convergence · Worst-case complexity

Mathematics Subject Classification 49J53 · 90C30 · 58C15 · 49M37 · 90C56 · 68Q25

Communicated by José Mario Martínez.

G. N. Grapiglia (✉) · J. Yuan
Departamento de Matemática, Universidade Federal do Paraná,
Centro Politécnico, Cx. Postal 19.081, Curitiba, Paraná 81531-980, Brazil
e-mail: geovani_mat@outlook.com

J. Yuan
e-mail: jin@ufpr.br

G. N. Grapiglia
The Capes Foundation, Ministry of Education of Brazil, Cx. Postal 250, Brasília,
Distrito Federal 70.040-020, Brazil

Y. Yuan
State Key Laboratory of Scientific/Engineering Computing,
Institute of Computational Mathematics and Scientific/Engineering Computing,
Academy of Mathematics and Systems Science, Chinese Academy of Sciences,
Zhong Guan Cun Donglu 55, Beijing 100190, People's Republic of China
e-mail: yyx@lsec.cc.ac.cn

Published online: 10 December 2014

1 Introduction

We consider the composite nonsmooth optimization (NSO) problem

$$\min_{x\in\mathbb{R}^n} \Phi(x) \equiv f(x) + h(c(x)), \tag{1}$$

where $f : \mathbb{R}^n \rightarrow \mathbb{R}$ and $c : \mathbb{R}^n \rightarrow \mathbb{R}^r$ are continuously differentiable and $h : \mathbb{R}^r \rightarrow \mathbb{R}$ is convex. The formulation of (1) with $f = 0$ includes problems of finding feasible points of nonlinear systems of inequalities (where $h(c) \equiv \|c^+\|_p$, with $c_i^+ = \max\{c_i, 0\}$ and $1 \leq p \leq +\infty$), finite minimax problems (where $h(c) \equiv \max_{1\leq i\leq r} c_i$), and best L_1, L_2 and L_∞ approximation problems (where $h(c) \equiv \|c\|_p$, $p = 1, 2, +\infty$). Another interesting example of problem (1) is the minimization of the exact penalty function

$$\Phi(x, \sigma) = f(x) + \sigma\|c(x)\|, \tag{2}$$

associated to the equality-constrained optimization problem

$$\begin{aligned} &\min \ f(x),\\ &\text{s.t. } c(x) = 0. \end{aligned} \tag{3}$$

There are several derivative-based algorithms to solve problem (1), which can be classified into two main groups.[1] The first group consists of methods for general nonsmooth optimization, such as subgradient methods (Shor 1978), bundle methods (Lemaréchal 1978), and cutting plane methods (Kelly 1960). If the function Φ satisfies suitable conditions, these methods are convergent in some sense. But they do not exploit the form of the problem, neither the convexity of h nor the differentiability of f and c. In contrast, the second group of methods is composed by methods in which the smooth substructure of the problem and the convexity of h are exploited. Notable algorithms in this group are those proposed by Fletcher (1982a, b), Powell (1983), Yuan (1985a), Bannert (1994) and Cartis et al. (2011b).

An essential feature of the algorithms mentioned above is that they require a subgradient of Φ or a gradient vector of f and a Jacobian matrix of c at each iteration. However, there are many practical problems where the derivative information is unreliable or unavailable (see, e.g., Conn et al. 2009a). In these cases, derivative-free algorithms become attractive, since they do not use (sub-)gradients explicitly.

In turn, derivative-free optimization methods can be distinguished into three groups. The first one is composed by direct-search methods, such as the Nelder–Mead method (Nelder and Mead 1965), the Hooke–Jeeves pattern search method (Hooke and Jeeves 1961) and the mesh adaptive direct-search (MADS) method (Audet and Dennis 2006). The second group consists of methods based on finite-differences, quasi-Newton algorithms or conjugate direction algorithms, whose examples can be found in Mifflin (1975), Stewart (1967), Greenstadt (1972) and Brent (1973). Finally, the third group is composed by methods based on interpolation models of the objective function, such as those proposed by Winfield (1973), Powell (2006) and Conn et al. (2009b).

Within this context, in this paper, we propose a globally convergent derivative-free trust-region algorithm for solving problem (1) when the function h is known and the gradient vectors of f and the Jacobian matrices of c are not available. The design of our algorithm is strongly based on the derivative-free trust-region algorithm proposed by Conn et al. (2009b) for unconstrained smooth optimization, and on the trust-region algorithms proposed by Fletcher (1982a), Powell (1983), Yuan (1985a) and Cartis et al. (2011b) for composite

[1] By derivative-based algorithm, we mean an algorithm that uses gradients or subgradients.

NSO. To the best of our knowledge, this work is the first one to present a derivative-free trust-region algorithm with global convergence results for the composite NSO problem, in which the structure of the problem is exploited.[2] We also investigate the worst-case function-evaluation complexity of the proposed algorithm. Specializing the complexity result for the minimization of (2), we obtain a complexity bound for solving equality-constrained optimization problems. At the end, we also present preliminary numerical results for minimax problems.

The paper is organized as follows. In Sect. 2, some definitions and results are reviewed and provided. The algorithm is presented in Sect. 3. Global convergence is treated in Sect. 4, while worst-case complexity bounds are given in Sects. 5 and 6. Finally, the numerical results are presented in Sect. 7.

2 Preliminaries

Throughout this paper, we shall consider the following concept of criticality.

Definition 1 (Yuan 1985b, page 271) A point x^* is said to be a critical point of Φ given in (1) if

$$f(x^*) + h(c(x^*)) \leq f(x^*) + \nabla f(x^*)^{\mathrm{T}} s + h(c(x^*) + J_c(x^*)s), \quad \forall s \in \mathbb{R}^n, \tag{4}$$

where J_c denotes the Jacobian of c.

Based on the above definition, for each $x \in \mathbb{R}^n$, define

$$l(x, s) \equiv f(x) + \nabla f(x)^{\mathrm{T}} s + h(c(x) + J_c(x)s), \quad \forall s \in \mathbb{R}^n, \tag{5}$$

and, for all $r > 0$, let

$$\psi_r(x) \equiv l(x, 0) - \min_{\|s\| \leq r} l(x, s). \tag{6}$$

Following Cartis et al. (2011b), we shall use the quantity $\psi_1(x)$ as a criticality measure for Φ. This choice is justified by the lemma below.

Lemma 1 *Let $\psi_1 : \mathbb{R}^n \to \mathbb{R}$ be defined by* (6). *Suppose that $f : \mathbb{R}^n \to \mathbb{R}$ and $c : \mathbb{R}^n \to \mathbb{R}^r$ are continuously differentiable, and that $h : \mathbb{R}^r \to \mathbb{R}$ is convex and globally Lipschitz. Then,*

(a) ψ_1 *is continuous*;
(b) $\psi_1(x) \geq 0$ *for all* $x \in \mathbb{R}^n$; *and*
(c) x^* *is a critical point of* $\Phi \iff \psi_1(x^*) = 0$.

Proof See Lemma 2.1 in Yuan (1985a). □

Our derivative-free trust-region algorithm will be based on the class of fully linear models proposed by Conn et al. (2009b). To define such class of models, let x_0 be the initial iterate

[2] For the finite minimax problem, which is a particular case of (1), a derivative-free trust-region algorithm was proposed by Madsen (1975), where the convergence to a critical point was proved under the assumption that the sequence of points $\{x_k\}$ generated by the algorithm is convergent. On the other hand, if the function Φ is locally Lipschitz, then problem (1) can be solved by the trust-region algorithm of Qi and Sun (1994), which, in theory, may not explicitly depend on subgradients or directional derivatives. However, this algorithm does not exploit the structure of the problem.

and suppose that the new iterates do not increase the value of the objective function Φ. Then, it follows that

$$x_k \in L(x_0) \equiv \left\{x \in \mathbb{R}^n \,:\, \Phi(x) \leq \Phi(x_0)\right\}, \quad \text{for all } k. \tag{7}$$

In this case, if the sampling to form the models is restricted to the closed balls $B[x_k; \Delta_k]$, and Δ_k is supposed bounded above by a constant $\bar{\Delta} > 0$, then Φ is only evaluated within the set

$$L_{enl}(x_0) = \bigcup_{x \in L(x_0)} B[x; \bar{\Delta}]. \tag{8}$$

Now, considering the sets $L(x_0)$ and $L_{enl}(x_0)$, we have the following definition.

Definition 2 (Conn et al. 2009b, Definition 3.1) Assume that $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable and that its gradient ∇f is Lipschitz continuous on $L_{enl}(x_0)$. A set of model functions $M = \left\{p : \mathbb{R}^n \to \mathbb{R} \,|\, p \in C^1\right\}$ is called a fully linear class of models if:

1. There exist constants $\kappa_{jf}, \kappa_f, \kappa_{lf} > 0$ such that for any $x \in L(x_0)$ and $\Delta \in (0, \bar{\Delta}]$ there exists a model function $p \in M$, with Lipschitz continuous gradient ∇p and corresponding Lipschitz constant bounded above by κ_{lf}, and such that

$$\|\nabla f(y) - \nabla p(y)\| \leq \kappa_{jf} \Delta, \quad \forall y \in B[x; \Delta], \tag{9}$$

   and

$$|f(y) - p(y)| \leq \kappa_f \Delta^2, \quad \forall y \in B[x; \Delta]. \tag{10}$$

   Such a model p is called fully linear on $B[x; \Delta]$.
2. For this class M, there exists an algorithm called "model-improvement" algorithm, that in a finite, uniformly bounded (with respect to x and Δ) number of steps can either establish that a given model $p \in M$ is fully linear on $B[x; \Delta]$, or find a model $\tilde{p} \in M$ that is fully linear on $B[x; \Delta]$.

Remark 1 Interpolation and regression linear models of the form

$$p(y) = c + g^{\mathrm{T}} y \tag{11}$$

are examples of fully linear models when the set of sample points is chosen in a convenient way (see Section 2.4 in Conn et al. 2009a). Furthermore, under some conditions, one can also prove that (interpolation or regression) quadratic models of the form

$$p(y) = c + g^{\mathrm{T}} y + \frac{1}{2} y^{\mathrm{T}} H y \quad (H \in \mathbb{R}^{n \times n} \text{ symmetric}) \tag{12}$$

are fully linear models (see Section 6.1 in Conn et al. 2009a). On the other hand, Algorithms 6.3 and 6.5 in Conn et al. (2009a) are examples of model-improvement algorithms (see Sections 6.2 and 6.3 in Conn et al. 2009a).

Remark 2 Let $c : \mathbb{R}^n \to \mathbb{R}^r$ be continuously differentiable with Jacobian function J_c Lipschitz continuous on $L_{enl}(x_0)$. If $q_i : \mathbb{R}^n \to \mathbb{R}$ is a fully linear model of $c_i : \mathbb{R}^n \to \mathbb{R}$ on $B[x; \Delta]$ for each $i = 1, \ldots, r$, then there exist constants $\kappa_{jc}, \kappa_c, \kappa_{lc} > 0$ such that the function $q \equiv (q_1, \ldots, q_r) : \mathbb{R}^n \to \mathbb{R}^r$ satisfies the inequalities

$$\|J_c(y) - J_q(y)\| \leq \kappa_{jc} \Delta, \quad \forall y \in B[x; \Delta], \tag{13}$$

and

$$\|c(y) - q(y)\| \leq \kappa_c \Delta^2, \quad \forall y \in B[x; \Delta]. \tag{14}$$

Moreover, the Jacobian J_q is Lipschitz continuous, and the corresponding Lipschitz constant is bounded by κ_{lc}. In this case, we shall say that q is a fully linear model of c on $B[x; \Delta]$ with respect to constants κ_{jc}, κ_c and κ_{lc}.

The next lemma establishes that if a model q is fully linear on $B[x; \Delta^*]$ with respect to some constants κ_{jc}, κ_c and κ_{lc}, then it is also fully linear on $B[x; \Delta]$ for any $\Delta \in [\Delta^*, \bar{\Delta}]$ with the same constants.

Lemma 2 *Consider a function $c : \mathbb{R}^n \to \mathbb{R}^r$ satisfying the assumptions in Remark 2. Given $x \in L(x_0)$ and $\Delta^* \leq \bar{\Delta}$, suppose that $q : \mathbb{R}^n \to \mathbb{R}^r$ is a fully linear model of c on $B[x; \Delta^*]$ with respect to constants κ_{jc}, κ_c and κ_{lc}. Assume also, without loss of generality, that κ_{jc} is no less than the sum of κ_{lc} and the Lipschitz constant of J_c. Then, q is fully linear on $B[x; \Delta]$, for any $\Delta \in [\Delta^*, \bar{\Delta}]$, with respect to the same constants κ_{jc}, κ_c and κ_{lc}.*

Proof It follows by the same argument used in the proof of Lemma 3.4 in Conn et al. (2009b). □

From here, consider the following assumptions:

A1 The functions $f : \mathbb{R}^n \to \mathbb{R}$ and $c : \mathbb{R}^n \to \mathbb{R}^r$ are continuously differentiable.
A2 The gradient of f, $\nabla f : \mathbb{R}^n \to \mathbb{R}^n$, and the Jacobian of c, $J_c : \mathbb{R}^n \to \mathbb{R}^{r \times n}$, are globally Lipschitz on $L_{enl}(x_0)$, with Lipschitz constants L_f and L_c, respectively.
A3 The function $h : \mathbb{R}^r \to \mathbb{R}$ is convex and globally Lipschitz continuous, with Lipschitz constant L_h.

Given functions $p : \mathbb{R}^n \to \mathbb{R}$ and $q : \mathbb{R}^n \to \mathbb{R}^r$ continuously differentiable, for each $x \in \mathbb{R}^n$ define

$$\tilde{l}(x, s) = f(x) + \nabla p(x)^{\mathrm{T}} s + h(c(x) + J_q(x)s), \quad \forall s \in \mathbb{R}^n. \tag{15}$$

and, for all $r > 0$, let

$$\eta_r(x) \equiv \tilde{l}(x, 0) - \min_{\|s\| \leq r} \tilde{l}(x, s). \tag{16}$$

The theorem below describes the relation between $\psi_1(x)$ and $\eta_1(x)$ when p and q are fully linear models of f and c around x.

Theorem 1 *Suppose that* A1–A3 *hold. Assume that $p : \mathbb{R}^n \to \mathbb{R}$ is a fully linear model of f with respect to constants κ_{jf}, κ_f and κ_{lf}, and that $q : \mathbb{R}^n \to \mathbb{R}^r$ is a fully linear model of c with respect to constants κ_{jc}, κ_c and κ_{lc}, both on the ball $B[x; \Delta]$. Then,*

$$|\psi_1(y) - \eta_1(y)| \leq \kappa_s \Delta, \quad \forall y \in B[x; \Delta], \tag{17}$$

with $\kappa_s = \kappa_{jf} + L_h \kappa_{jc}$.

Proof Let $y \in B[x; \Delta]$. Since p and q are fully linear models of f and c, respectively, on the ball $B[x; \Delta]$, it follows that

$$\|\nabla f(y) - \nabla p(y)\| \leq \kappa_{jf} \Delta \quad \text{and} \quad \|J_c(y) - J_q(y)\| \leq \kappa_{jc} \Delta. \tag{18}$$

Consider $\tilde{s} \in B[0; 1]$ such that

$$\min_{\|s\| \leq 1} l(y, s) = l(y, \tilde{s}). \tag{19}$$

Then, from A3, (18) and (19), it follows that

$$\begin{aligned}\psi_1(y)-\eta_1(y) &= \left(l(y,0)-\min_{\|s\|\leq 1} l(y,s)\right)-\left(\tilde{l}(y,0)-\min_{\|s\|\leq 1}\tilde{l}(y,s)\right)\\ &= \min_{\|s\|\leq 1}\tilde{l}(y,s)-\min_{\|s\|\leq 1} l(y,s)\\ &= \min_{\|s\|\leq 1}\tilde{l}(y,s)-l(y,\tilde{s})\\ &\leq \tilde{l}(y,\tilde{s})-l(y,\tilde{s})\\ &= (\nabla p(y)-\nabla f(y))^{\mathrm{T}}\tilde{s}+\left[h(c(y)+J_q(y)\tilde{s})-h(c(y)+J_c(y)\tilde{s})\right]\\ &\leq \|\nabla p(y)-\nabla f(y)\|+L_h\|J_q(y)-J_c(y)\|\\ &= \left(\kappa_{jf}+L_h\kappa_{jc}\right)\Delta. \end{aligned} \tag{20}$$

Similarly, considering $\bar{s}\in B[0;1]$ such that

$$\min_{\|s\|\leq 1}\tilde{l}(y,s)=\tilde{l}(y,\bar{s}), \tag{21}$$

we obtain the inequality

$$\eta_1(y)-\psi_1(y)\leq\left(\kappa_{jf}+L_h\kappa_{jc}\right)\Delta. \tag{22}$$

Hence, from (20) and (22), we conclude that (17) holds. □

3 Algorithm

Considering the theory discussed above, in this section, we present an algorithm to solve (1) without derivatives. The algorithm is an adaptation of Algorithm 4.1 in Conn et al. (2009b) for unconstrained smooth optimization, and contains elements of the trust-region algorithms of Fletcher (1982a), Powell (1983), Yuan (1985a) and Cartis et al. (2011b) for composite NSO.

Algorithm 1 (Derivative-Free Trust-Region Algorithm for Composite NSO)

Step 0 Choose a class of fully linear models for f and c (e.g., linear interpolation models) and a corresponding model-improvement algorithm (e.g., Algorithm 6.3 in Conn et al. 2009a). Choose $x_0\in\mathbb{R}^n$, $\bar{\Delta}>0$, $\Delta_0^*\in(0,\bar{\Delta}]$, $H_0\in\mathbb{R}^{n\times n}$ symmetric, $0\leq\alpha_0\leq\alpha_1<1$ (with $\alpha_1\neq 0$), $0<\gamma_1<1<\gamma_2$, $\epsilon_c>0$, $\mu>\beta>0$ and $\omega\in(0,1)$. Consider a model $p_0^*(x_0+d)$ for f (with gradient at $d=0$ denoted by g_0^*) and a model $q_0^*(x_0+d)$ for c (with Jacobian matrix at $d=0$ denoted by A_0^*). Set $k:=0$.

Step 1 Compute

$$\eta_1^*(x_k)=l^*(x_k,0)-l^*(x_k,s_k),$$

where

$$s_k=\arg\min_{\|s\|\leq 1} l^*(x_k,s), \tag{23}$$

and

$$l^*(x_k,s)=f(x_k)+(g_k^*)^{\mathrm{T}}s+h(c(x_k)+A_k^*s).$$

If $\eta_1^*(x_k)>\epsilon_c$, then set $\eta_1(x_k)=\eta_1^*(x_k)$, $p_k=p_k^*$ $(g_k=g_k^*)$, $q_k=q_k^*$ $(A_k=A_k^*)$, $\Delta_k=\Delta_k^*$, and go to Step 3.

Step 2 Call the model-improvement algorithm to verify if the models p_k^* and q_k^* are fully linear on $B\left[x_k; \Delta_k^*\right]$. If $\Delta_k^* \leq \mu\eta_1^*(x_k)$ and the models p_k^* and q_k^* are fully linear on $B[x_k; \Delta_k^*]$, set $\eta_1(x_k) = \eta_1^*(x_k)$, $p_k = p_k^*$ $(g_k = g_k^*)$, $q_k = q_k^*$ $(A_k = A_k^*)$ and $\Delta_k = \Delta_k^*$. Otherwise, apply Algorithm 2 (described below) to construct a model $\tilde{p}_k(x_k+d)$ for f (with gradient at $d=0$ denoted by $\tilde{g}_k$) and a model $\tilde{q}_k(x_k+d)$ for c (with Jacobian matrix at $d=0$ denoted by $\tilde{A}_k$), both fully linear (for some constants which remain the same for all iterations of Algorithm 1) on the ball $B[x_k; \tilde{\Delta}_k]$, for some $\tilde{\Delta}_k \in (0, \mu\tilde{\eta}_1(x_k)]$, where $\tilde{\Delta}_k$ and $\tilde{\eta}_1(x_k)$ are given by Algorithm 2. In such case, set $\eta_1(x_k) = \tilde{\eta}_1(x_k)$, $p_k = \tilde{p}_k$ $(g_k = \tilde{g}_k)$, $q_k = \tilde{q}_k$ $(A_k = \tilde{A}_k)$ and

$$\Delta_k = \min\left\{\max\left\{\tilde{\Delta}_k, \beta\tilde{\eta}_1(x_k)\right\}, \Delta_k^*\right\}. \tag{24}$$

Step 3 Let D_k^* be the set of solutions of the subproblem

$$\min_{d\in\mathbb{R}^n} \quad m_k(x_k+d) \equiv f(x) + g_k^{\mathrm{T}}d + h\left(c(x_k)+A_kd\right) + \frac{1}{2}d^{\mathrm{T}}H_kd \tag{25}$$

$$\text{s.t.} \quad \|d\| \leq \Delta_k \tag{26}$$

Compute a step d_k for which $\|d_k\| \leq \Delta_k$ and

$$m_k(x_k) - m_k(x_k+d_k) \geq \alpha_2\left[m_k(x_k) - m_k(x_k+d_k^*)\right], \tag{27}$$

for some $d_k^* \in D_k^*$, where $\alpha_2 \in (0,1)$ is a constant independent of k.

Step 4 Compute $\Phi(x_k+d_k)$ and define

$$\rho_k = \frac{\Phi(x_k) - \Phi(x_k+d_k)}{m_k(x_k) - m_k(x_k+d_k)}.$$

If $\rho_k \geq \alpha_1$ or if both $\rho_k \geq \alpha_0$ and the models p_k and q_k are fully linear on $B[x_k; \Delta_k]$, then $x_{k+1} = x_k + d_k$ and the models are updated to include the new iterate into the sample set, resulting in new models $p_{k+1}^*(x_{k+1}+d)$ (with gradient at $d=0$ denoted by g_{k+1}^*) and $q_{k+1}^*(x_{k+1}+d)$ (with Jacobian matrix at $d=0$ denoted by A_{k+1}^*). Otherwise, set $x_{k+1} = x_k$, $p_{k+1}^* = p_k$ $(g_{k+1}^* = g_k)$ and $q_{k+1}^* = q_k$ $(A_{k+1}^* = A_k)$.

Step 5 If $\rho_k < \alpha_1$, use the model-improvement algorithm to certify if p_k and q_k are fully linear models on $B[x_k; \Delta_k]$. If such a certificate is not obtained, we declare that p_k or q_k is not certifiably fully linear (CFL) and make one or more improvement steps. Define p_{k+1}^* and q_{k+1}^* as the (possibly improved) models.

Step 6 Set

$$\Delta_{k+1}^* \in \begin{cases} \left[\Delta_k, \min\left\{\gamma_2\Delta_k, \bar{\Delta}\right\}\right] & \text{if } \rho_k \geq \alpha_1, \\ \{\gamma_1\Delta_k\} & \text{if } \rho_k < \alpha_1 \text{ and } p_k \\ & \quad \text{and } q_k \text{ are fully linear,} \\ \{\Delta_k\} & \text{if } \rho_k < \alpha_1 \text{ and } p_k \\ & \quad \text{or } q_k \text{ is not CFL.} \end{cases} \tag{28}$$

Step 7 Generate H_{k+1}, set $k := k+1$ and go to Step 1.

Remark 3 The matrix H_k in (25) approximates the second-order behavior of f and c around x_k. For example, as suggested by equation (1.5) in Yuan (1985b), one could use $H_k = \nabla^2 p_k(x_k) + \sum_{i=1}^r \nabla^2(q_k)_i(x_k)$. Moreover, if h is a polyhedral function and $\|.\| = \|.\|_\infty$, then subproblem (25)–(26) reduces to a quadratic programming problem, which can be solved by standard methods.

In the above algorithm, the iterations for which $\rho_k \geq \alpha_1$ are said to be *successful iterations*; the iterations for which $\rho_k \in [\alpha_0, \alpha_1)$ and p_k and q_k are fully linear are said to be *acceptable iterations*; the iterations for which $\rho_k < \alpha_1$ and p_k or q_k is not certifiably fully linear are said to be *model-improving iterations* and; the iterations for which $\rho_k < \alpha_0$ and p_k and q_k are fully linear are said to be *unsuccessful iterations*.

Below we describe the Algorithm 2 used in Step 2, which is an adaptation of the Algorithm 4.2 in Conn et al. (2009b).

Algorithm 2 *This algorithm is only applied if* $\eta_1^*(x_k) \leq \epsilon_c$ *and at least one of the following holds: the model* p_k^* *or the model* q_k^* *is not CFL on* $B\left[x_k; \Delta_k^*\right]$ *or* $\Delta_k^* > \mu\eta_1^*(x_k)$. *The constant* $\omega \in (0, 1)$ *is chosen at Step 0 of Algorithm 1.*

Initialization: Set $p_k^{(0)} = p_k^*$, $q_k^{(0)} = q_k^*$ and $i = 0$.

Repeat Set $i := i + 1$.

Use the model-improvement algorithm to improve the previous models $p_k^{(i-1)}$ and $q_k^{(i-1)}$ until they become fully linear on $B\left[x_k; \omega^{i-1}\Delta_k^*\right]$ (by Definition 2, this can be done in a finite, uniformly bounded number of steps of the model-improvement algorithm). Denote the new models by $p_k^{(i)}$ and $q_k^{(i)}$. Set $\tilde{\Delta}_k = \omega^{i-1}\Delta_k^*$, $\tilde{p}_k = p_k^{(i)}$, $\tilde{g}_k = \nabla\tilde{p}_k(x_k)$, $\tilde{q}_k = q_k^{(i)}$ and $\tilde{A}_k = J_{\tilde{q}_k}(x_k)$. Compute

$$\eta_1^{(i)}(x_k) = \tilde{l}(x_k, 0) - \tilde{l}(x_k, \tilde{s}_k),$$

where

$$\tilde{s}_k = \arg\min_{\|s\|\leq 1} \tilde{l}(x_k, s), \tag{29}$$

and

$$\tilde{l}(x_k, s) \equiv f(x_k) + (\tilde{g}_k)^{\mathrm{T}} s + h(c(x_k) + \tilde{A}_k s).$$

Set $\tilde{\eta}_1(x_k) = \eta_1^{(i)}(x_k)$.

Until $\tilde{\Delta}_k \leq \mu\eta_1^{(i)}(x_k)$.

Remark 4 When $h \equiv 0$, then Algorithms 1 and 2 are reduced to the corresponding algorithms in Conn et al. (2009b) for minimizing f without derivatives.

Remark 5 If Step 2 is executed in Algorithm 1, then the models p_k and q_k are fully linear on $B[x_k; \tilde{\Delta}_k]$ with $\tilde{\Delta}_k \leq \Delta_k$. Hence, by Lemma 2, p_k and q_k are also fully linear on $B[x_k; \Delta_k]$ (as well on $B[x_k; \mu\eta_1(x_k)]$).

4 Global convergence analysis

In this section, we shall prove the weak global convergence of Algorithm 1. Thanks to Theorem 1, it can be done by a direct adaptation of the analysis presented by Conn et al. (2009b). In fact, the proof of most of the results below follows by simply changing the criticality measures (g_k and $\nabla f(x_k)$ in Conn et al. (2009b) to $\eta_1(x_k)$ and $\psi_1(x_k)$, respectively). Thus, in these cases, we only indicate the proof of the corresponding result in Conn et al. (2009b).

The first result says that unless the current iterate is a critical point, Algorithm 2 will not loop infinitely in Step 2 of Algorithm 1.

Lemma 3 *Suppose that* A1–A3 *hold. If* $\psi_1(x_k) \neq 0$, *then Algorithm* 2 *will terminate in a finite number of repetitions.*

Proof See the proof of Lemma 5.1 in Conn et al. (2009b). □

The next lemma establishes the relation between $\eta_r(x)$ and $\eta_1(x)$.

Lemma 4 *Suppose that* A3 *holds and let* $r > 0$. *Then, for all* x

$$\eta_r(x) \geq \min\{1, r\}\, \eta_1(x). \tag{30}$$

Proof It follows as in the proof of Lemma 2.1 in Cartis et al. (2011b). □

Lemma 5 *Suppose that* A3 *holds. Then, there exists a constant* $\kappa_d > 0$ *such that the inequality*

$$m_k(x_k) - m_k(x_k + d_k) \geq \kappa_d \eta_1(x_k) \min\left\{\Delta_k, \frac{\eta_1(x_k)}{1 + \|H_k\|}\right\}. \tag{31}$$

holds for all k.

Proof From inequality (27) and Lemma 4, it follows as in the proof of Lemma 11 in Grapiglia et al. (2014). □

For the rest of the paper, we consider the following additional assumptions:

A4 There exists a constant $\kappa_H > 0$ such that $\|H_k\| \leq \kappa_H$ for all k.
A5 The sequence $\{\Phi(x_k)\}$ is bounded below by Φ_{low}.

Lemma 6 *Suppose that* A1–A4 *hold. If* p_k *and* q_k *are fully linear models of* f *and* c, *respectively, on the ball* $B[x_k; \Delta_k]$, *then*

$$\Phi(x_k + d_k) - m_k(x_k + d_k) \leq \kappa_m \Delta_k^2, \tag{32}$$

where $\kappa_m = (L_f + 2\kappa_{jf} + L_h L_c + 2L_h \kappa_{jc} + \kappa_H)/2$.

Proof In fact, from Assumptions A1–A4, it follows that

$$\begin{aligned}
\Phi(x_k + d_k) - m_k(x_k + d_k) &= f(x_k + d_k) + h(c(x_k + d_k)) - f(x_k) - \nabla p_k(x_k)^{\mathrm{T}} d_k \\
&\quad - h(c(x_k) + J_{q_k}(x_k) d_k) - \frac{1}{2} d_k^{\mathrm{T}} H_k d_k \\
&\leq f(x_k + d_k) - f(x_k) - \nabla f(x_k)^{\mathrm{T}} d_k + (\nabla f(x_k) - \nabla p_k(x_k))^{\mathrm{T}} d_k \\
&\quad + h(c(x_k + d_k)) - h(c(x_k) + J_{q_k}(x_k) d_k) + \frac{1}{2} \kappa_H \|d_k\|^2 \\
&\leq \frac{L_f}{2} \Delta_k^2 + \kappa_{jf} \Delta_k^2 + L_h \|c(x_k + d_k) - c(x_k) - J_c(x_k) d_k\| \\
&\quad + L_h \|(J_c(x_k) - J_{q_k}(x_k)) d_k\| + \frac{1}{2} \kappa_H \Delta_k^2 \\
&\leq \frac{1}{2}\left(L_f + 2\kappa_{jf} + L_h L_c + 2L_h \kappa_{jc} + \kappa_H\right) \Delta_k^2.
\end{aligned}$$

□

The proof of the next lemma is based on the proof of Lemma 5.2 in Conn et al. (2009b).

Lemma 7 *Suppose that* A1–A4 *hold. If p_k and q_k are fully linear models of f and c, respectively, on the ball $B[x_k; \Delta_k]$, and*

$$\Delta_k \leq \min\left\{\frac{1}{1+\kappa_H}, \frac{\kappa_d(1-\alpha_1)}{\kappa_m}\right\}\eta_1(x_k), \tag{33}$$

then, the k-th iteration is successful.

Proof Since $\|H_k\| \leq \kappa_H$ and $\Delta_k \leq \eta_1(x_k)/(1+\kappa_H)$, it follows from Lemma 5 that

$$m_k(x_k) - m_k(x_k+d_k) \geq \kappa_d \eta_1(x_k)\Delta_k. \tag{34}$$

Then, by Lemma 6 and (33), we have

$$\begin{aligned} 1-\rho_k &= \frac{(m_k(x_k)-m_k(x_k+d_k)) - (\Phi(x_k)-\Phi(x_k+d_k))}{m_k(x_k)-m_k(x_k+d_k)} \\ &= \frac{\Phi(x_k+d_k)-m_k(x_k+d_k)}{m_k(x_k)-m_k(x_k+d_k)} \\ &\leq \frac{\kappa_m \Delta_k}{\kappa_d \eta_1(x_k)} \\ &\leq 1-\alpha_1. \end{aligned} \tag{35}$$

Hence, $\rho_k \geq \alpha_1$, and consequently iteration k is successful. □

The next lemma gives a lower bound on Δ_k when $\eta_1(x_k)$ is bounded away from zero. Its proof is based on the proof of Lemma 5.3 in Conn et al. (2009b), and on the proof of the lemma on page 299 in Powell (1984).

Lemma 8 *Suppose that* A1–A4 *hold and let $\epsilon > 0$ such that $\eta_1(x_k) \geq \epsilon$ for all $k = 0, \ldots, j$, where $j \leq +\infty$. Then, there exists $\bar{\tau} > 0$ independent of k such that*

$$\Delta_k \geq \bar{\tau}, \quad \textit{for all } k = 0, \ldots, j. \tag{36}$$

Proof We shall prove (36) by induction over k with

$$\bar{\tau} = \min\left\{\beta\epsilon, \Delta_0^*, \frac{\gamma_1\epsilon}{1+\kappa_H}, \frac{\gamma_1\kappa_d(1-\alpha_1)\epsilon}{\kappa_m}\right\}. \tag{37}$$

From equality (24) in Step 2 of Algorithm 1, it follows that

$$\Delta_k \geq \min\left\{\beta\eta_1(x_k), \Delta_k^*\right\}, \quad \text{for all } k. \tag{38}$$

Hence, since $\eta_1(x_k) \geq \epsilon$ for $k = 0, \ldots, j$, we have

$$\Delta_k \geq \min\left\{\beta\epsilon, \Delta_k^*\right\}, \quad \text{for all } k = 0, \ldots, j. \tag{39}$$

In particular, (39) implies that (36) holds for $k = 0$.

Now, we assume that (36) is true for $k \in \{0, \ldots, j-1\}$ and prove it is also true for $k+1$. First, suppose that

$$\Delta_k \leq \min\left\{\frac{\epsilon}{1+\kappa_H}, \frac{\kappa_d(1-\alpha_1)\epsilon}{\kappa_m}\right\}. \tag{40}$$

Then, by Lemma 7 and Step 5, the k-th iteration is successful or model improving, and so by Step 6 $\Delta_{k+1}^* \geq \Delta_k$. Hence, (39), the induction assumption and (37) imply that

$$\Delta_{k+1} \geq \min\left\{\beta\epsilon, \Delta_{k+1}^*\right\} \geq \min\left\{\beta\epsilon, \Delta_k\right\} \geq \min\left\{\beta\epsilon, \bar{\tau}\right\} = \bar{\tau}. \tag{41}$$

Therefore, (36) is true for $k+1$. On the other hand, suppose that (40) is not true. Then, from (28), (39) and (37), it follows that

$$\begin{aligned} \Delta^*_{k+1} &\geq \gamma_1 \Delta_k > \min\left\{\frac{\gamma_1\epsilon}{1+\kappa_H}, \frac{\gamma_1\kappa_d(1-\alpha_1)\epsilon}{\kappa_m}\right\} \geq \bar{\tau} \\ &\Longrightarrow \Delta_{k+1} \geq \min\left\{\beta\epsilon, \Delta^*_{k+1}\right\} \geq \min\left\{\beta\epsilon, \bar{\tau}\right\} = \bar{\tau}. \end{aligned}$$

Hence, (36) holds for $k = 0, \ldots, j$. □

Lemma 9 *Suppose that* A1–A5 *hold. Then,*

$$\lim_{k\to+\infty} \Delta_k = 0. \tag{42}$$

Proof See Lemmas 5.4 and 5.5 in Conn et al. (2009b). □

The proof of the next result is based on the proof of Lemma 5.6 in Conn et al. (2009b).

Lemma 10 *Suppose that* A1–A5 *hold. Then,*

$$\liminf_{k\to+\infty} \eta_1(x_k) = 0. \tag{43}$$

Proof Suppose that (43) is not true. Then, there exists a constant $\epsilon > 0$ such that

$$\eta_1(x_k) \geq \epsilon, \quad \text{for all } k. \tag{44}$$

In this case, by Lemma 8, there exists $\bar{\tau} > 0$ such that $\Delta_k \geq \bar{\tau}$ for all k, contradicting Lemma 9.

The lemma below says that if a subsequence of $\{\eta_1(x_k)\}$ converges to zero, so does the corresponding subsequence of $\{\psi_1(x_k)\}$.

Lemma 11 *Suppose that* A1–A3 *hold. Then, for any sequence* $\{k_i\}$ *such that*

$$\lim_{i\to\infty} \eta_1(x_{k_i}) = 0, \tag{45}$$

it also holds that

$$\lim_{i\to\infty} \psi_1(x_{k_i}) = 0. \tag{46}$$

Proof See Lemma 5.7 in Conn et al. (2009b). □

Now, we can obtain the weak global convergence result.

Theorem 2 *Suppose that* A1–A5 *hold. Then,*

$$\liminf_{k\to+\infty} \psi_1(x_k) = 0. \tag{47}$$

Proof It follows directly from Lemmas10 and 11. □

By Theorem 2 and Lemma 1(a), at least one accumulation point of the sequence $\{x_k\}$ generated by Algorithm 1 is a critical point of Φ (in the sense of Definition 1).

5 Worst-case complexity analysis

In this section, we shall study the worst-case complexity of a slight modification of Algorithm 1 following closely the arguments of Cartis et al. (2011a, b). For that, we replace (28) by the rule

$$\Delta_{k+1}^{*} \in \begin{cases} \left[\Delta_k, \min\left\{\gamma_2\Delta_k, \bar{\Delta}\right\}\right] & \text{if } \rho_k \geq \alpha_1, \\ \{\gamma_1\Delta_k\} & \text{if } \rho_k < \alpha_1 \text{ and } p_k \\ & \quad \text{and } q_k \text{ are fully linear,} \\ \{\gamma_0\Delta_k\} & \text{if } \rho_k < \alpha_1 \text{ and } p_k \\ & \quad \text{or } q_k \text{ is not CFL,} \end{cases} \tag{48}$$

where $0 < \gamma_1 < \gamma_0 < 1 < \gamma_2$. Moreover, we consider $\epsilon_c = +\infty$ in Step 0. In this way, Step 2 will be called at all iterations and, consequently, the models p_k and q_k will be fully linear for all k.

Definition 3 Given $\epsilon \in (0, 1]$, a point $x^* \in \mathbb{R}^n$ is said to be an ϵ-critical point of Φ if $\psi_1(x^*) \leq \epsilon$.

Let $\{x_k\}$ be a sequence generated by Algorithm 1. Since the computation of $\psi_1(x_k)$ requires the gradient $\nabla f(x_k)$ and the Jacobian matrix $J_c(x_k)$ [recall (5) and (6)], in the derivative-free framework we cannot test directly whether an iterate x_k is ϵ-critical or not. One way to detect ϵ-criticality is to test $\eta_1(x_k)$ based on the following lemma.

Lemma 12 *Suppose that* A1–A3 *hold and let* $\epsilon \in (0, 1]$. *If*

$$\eta_1(x_k) \leq \frac{\epsilon}{(\kappa_s\mu + 1)} \equiv \epsilon_s, \tag{49}$$

then x_k is an ϵ-critical point of Φ.

Proof Since the models p_k and q_k are fully linear on a ball $B[x_k; \Delta_k]$ with $\Delta_k \leq \mu\eta_1(x_k)$, by Theorem 1, we have

$$|\psi_1(x_k) - \eta_1(x_k)| \leq \kappa_s\Delta_k \leq \kappa_s\mu\eta_1(x_k). \tag{50}$$

Consequently, from (49) and (50), it follows that

$$|\psi_1(x_k)| \leq |\psi_1(x_k) - \eta_1(x_k)| + |\eta_1(x_k)| \leq (\kappa_s\mu + 1)\eta_1(x_k) \leq \epsilon, \tag{51}$$

that is, x_k is an ϵ-critical point of Φ. □

Another way to detect ϵ-criticality is provided by Algorithm 2. The test is based on the lemma below.

Lemma 13 *Suppose that* A1–A3 *hold and let* $\epsilon \in (0, 1]$. *Denote by* $i_1 + 1$ *the number of times that the loop in Algorithm* 2 *is repeated. If*

$$i_1 \geq 1 + log\left(\frac{\epsilon}{(\kappa_s + \mu^{-1})\bar{\Delta}}\right)/log(\omega), \tag{52}$$

then the current iterate x_k is an ϵ-critical point of Φ.

Proof Let $I_1 = \{i \in \mathbb{N} \,|\, 0 < i \leq i_1\}$. Then, we must have

$$\mu\eta_1^{(i)}(x_k) < \omega^{i-1}\Delta_k^*, \tag{53}$$

for all $i \in I_1$. Since the models $p_k^{(i)}$ and $q_k^{(i)}$ were fully linear on $B[x_k; \omega^{i-1}\Delta_k^*]$, it follows from Theorem 1 that

$$|\psi_1(x_k) - \eta_1^{(i)}(x_k)| \leq \kappa_s \omega^{i-1}\Delta_k^*, \tag{54}$$

for each $i \in I_1$. Hence,

$$|\psi_1(x_k)| \leq |\psi_1(x_k) - \eta_1^{(i)}(x_k)| + |\eta_1^{(i)}(x_k)| \leq (\kappa_s + \mu^{-1})\omega^{i-1}\Delta_k^* \leq (\kappa_s + \mu^{-1})\omega^{i-1}\bar{\Delta}, \tag{55}$$

for all $i \in I_1$. On the other hand, since $\omega \in (0, 1)$, by (52), we have

$$\begin{aligned}
i_1 &\geq 1 + \log\left(\frac{\epsilon}{(\kappa_s + \mu^{-1})\bar{\Delta}}\right) / \log(\omega) \\
&\Longrightarrow i_1 - 1 \geq \log\left(\frac{\epsilon}{(\kappa_s + \mu^{-1})\bar{\Delta}}\right) / \log(\omega) \\
&\Longrightarrow (i_1 - 1)\log(\omega) \leq \log\left(\frac{\epsilon}{(\kappa_s + \mu^{-1})\bar{\Delta}}\right) \\
&\Longrightarrow \log(\omega^{i_1 - 1}) \leq \log\left(\frac{\epsilon}{(\kappa_s + \mu^{-1})\bar{\Delta}}\right) \\
&\Longrightarrow \omega^{i_1 - 1} \leq \frac{\epsilon}{(\kappa_s + \mu^{-1})\bar{\Delta}} \\
&\Longrightarrow (\kappa_s + \mu^{-1})\omega^{i_1 - 1}\bar{\Delta} \leq \epsilon.
\end{aligned} \tag{56}$$

Hence, from (55) and (56), it follows that $\psi_1(x_k) \leq \epsilon$, that is, x_k is an ϵ-critical point of Φ. □

In what follows, the worst-case complexity is defined as the maximum number of function evaluations necessary for criterion (49) to be satisfied. For convenience, we will consider the following notation:

$$S = \{k \geq 0 \,|\, k \text{ successful}\}, \tag{57}$$

$$S_j = \{k \leq j \,|\, k \in S\}, \quad \text{for each } j \geq 0, \tag{58}$$

$$U_j = \{k \leq j \,|\, k \notin S\} \quad \text{for each } j \geq 0, \tag{59}$$

$$S' = \{k \in S \,|\, \eta_1(x_k) > \epsilon_s = \epsilon_s(\epsilon)\}, \quad \epsilon > 0, \tag{60}$$

where S_j and U_j form a partition of $\{1, \ldots, j\}$, $|S_j|$, $|U_j|$ and $|S'|$ will denote the cardinality of these sets, and $\epsilon_s = \epsilon_s(\epsilon)$ is defined in (49). Furthermore, let $\tilde{S}$ be a generic index set such that

$$\tilde{S} \subseteq S', \tag{61}$$

and whose cardinality is denoted by $|\tilde{S}|$.

The next lemma provides an upper bound on the cardinality $|\tilde{S}|$ of a set $\tilde{S}$ satisfying (61).

Lemma 14 *Suppose that* A3 *and* A5 *hold. Given any* $\epsilon > 0$, *let* S' *and* $\tilde{S}$ *be defined in* (60) *and* (61), *respectively. Suppose that the successful iterates* x_k *generated by Algorithm* 1 *have the property that*

$$m_k(x_k) - m_k(x_k + d_k) \geq \theta_c \epsilon^p, \quad \textit{for all } k \in \tilde{S}, \tag{62}$$

where θ_c *is a positive constant independent of* k *and* ϵ, *and* $p > 0$. *Then,*

$$|\tilde{S}| \leq \left\lceil \kappa_p \epsilon^{-p} \right\rceil, \tag{63}$$

where $\kappa_p \equiv (\Phi(x_0) - \Phi_{\text{low}})/(\alpha_1 \theta_c)$.

Proof It follows as in the proof of Theorem 2.2 in Cartis et al. (2011a). □

The next lemma gives a lower bound on Δ_k when $\eta_1(x_k)$ is bounded away from zero.

Lemma 15 *Suppose that* A1–A4 *hold and let* $\epsilon \in (0, 1]$ *such that* $\eta_1(x_k) \geq \epsilon_s$ *for all* $k = 0, \ldots, j$, *where* $j \leq +\infty$ *and* $\epsilon_s = \epsilon_s(\epsilon)$ *is defined in* (49). *Then, there exists* $\tau > 0$ *independent of* k *and* ϵ *such that*

$$\Delta_k \geq \tau \epsilon, \quad \textit{for all } k = 0, \ldots, j. \tag{64}$$

Proof It follows from Lemma 8. □

Now, we can obtain an iteration complexity bound for Algorithm 1. The proof of this result is based on the proof of Theorem 2.1 and Corollary 3.4 in Cartis et al. (2011a).

Theorem 3 *Suppose that* A1–A5 *hold. Given any* $\epsilon \in (0, 1]$, *assume that* $\eta_1(x_0) > \epsilon_s$ *and let* $j_1 \leq +\infty$ *be the first iteration such that* $\eta_1(x_{j_1+1}) \leq \epsilon_s$, *where* $\epsilon_s = \epsilon_s(\epsilon)$ *is defined in* (49). *Then, Algorithm* 1 *with rule* (48) *and* $\epsilon_c = +\infty$ *takes at most*

$$L_1^s \equiv \left\lceil \kappa_c^s \epsilon^{-2} \right\rceil \tag{65}$$

successful iterations to generate $\eta_1(x_{j_1+1}) \leq \epsilon_s$ *and consequently (by Lemma* 12*)* $\psi_1(x_{j_1+1}) \leq \epsilon$, *where*

$$\kappa_c^s \equiv (\Phi(x_0) - \Phi_{\text{low}})/(\alpha_1 \theta_c), \quad \theta_c = \kappa_d \min\{\tau, 1/(1 + \kappa_H)\}/(\kappa_s \mu + 1)^2. \tag{66}$$

Furthermore,

$$j_1 \leq \left\lceil \kappa_w \epsilon^{-2} \right\rceil \equiv L_1, \tag{67}$$

and Algorithm 1 *takes at most* L_1 *iterations to generate* $\psi_1(x_{j_1+1}) \leq \epsilon$, *where*

$$\kappa_w \equiv \left(1 - \frac{\log(\gamma_2^{-1})}{\log(\gamma_2^{-1})}\right) \kappa_c^s + \frac{(\kappa_s \mu + 1)\Delta_0^*}{\tau \log(\gamma_2^{-1})}. \tag{68}$$

Proof The definition of j_1 in the statement of the theorem implies that

$$\eta_1(x_k) > \epsilon_s, \quad \text{for } k = 0, \ldots, j_1. \tag{69}$$

Thus, by Lemma 5, Assumption A4, Lemma 15 and the definition of ϵ_s in (49), we obtain

$$\begin{aligned} m_k(x_k) - m_k(x_k + d_k) &\geq \kappa_d \epsilon_s \min\left\{\Delta_k, \frac{\epsilon_s}{1 + \kappa_H}\right\} \\ &\geq \frac{\kappa_d}{(\kappa_s \mu + 1)^2} \epsilon \min\left\{\tau \epsilon, \frac{\epsilon}{1 + \kappa_H}\right\} \end{aligned}$$

$$\begin{aligned} &= \frac{\kappa_d}{(\kappa_s\mu+1)^2}\min\{\tau, 1/(1+\kappa_H)\}\epsilon^2 \\ &= \theta_c\epsilon^2, \quad \text{for } k=1,\ldots,j_1, \end{aligned} \tag{70}$$

where θ_c is defined by (66). Now, with $j = j_1$ in (58) and (59), Lemma 14 with $\tilde{S} = S_{j_1}$ and $p = 2$ provides the complexity bound

$$|S_{j_1}| \le L_1^s, \tag{71}$$

where L_1^s is defined by (65).

On the other hand, from rule (48) and Lemma 15, it follows that

$$\begin{aligned} \Delta_{k+1}^* &\le \gamma_2\Delta_k \le \gamma_2\Delta_k^*, \quad \text{if } k\in S_{j_1}, \\ \Delta_{k+1}^* &\le \gamma_0\Delta_k \le \gamma_0\Delta_k^*, \quad \text{if } k\in U_{j_1}, \\ \Delta_k^* &\ge \Delta_k \ge \frac{\tau}{\kappa_s\mu+1}\epsilon, \quad \text{for } k=0,\ldots,j_1. \end{aligned}$$

Thus, considering $\omega_k \equiv 1/\Delta_k^*$, we have

$$c_1\omega_k \le \omega_{k+1}, \quad \text{if } k\in S_{j_1}, \tag{72}$$

$$c_2\omega_k \le \omega_{k+1}, \quad \text{if } k\in U_{j_1}, \tag{73}$$

$$\omega_k \le \bar{\omega}\epsilon^{-1}, \quad \text{for } k=0,\ldots,j_1, \tag{74}$$

where $c_1 = \gamma_2^{-1} \in (0,1]$, $c_2 = \gamma_0^{-1} > 1$ and $\bar{\omega} = (\kappa_s\mu+1)/\tau$. From (72) and (73), we deduce inductively

$$\omega_0 c_1^{|S_{j_1}|} c_2^{|U_{j_1}|} \le \omega_{j_1}.$$

Hence, from (74), it follows that

$$c_1^{|S_{j_1}|} c_2^{|U_{j_1}|} \le \frac{\bar{\omega}\epsilon^{-1}}{\omega_0}, \tag{75}$$

and so, taking logarithm on both sides, we get

$$|U_{j_1}| \le \left\lceil -\frac{\log(c_1)}{\log(c_2)}|S_{j_1}| + \frac{\bar{\omega}}{\omega_0\log(c_2)}\epsilon^{-1}\right\rceil. \tag{76}$$

Finally, since $j_1 = |S_{j_1}| + |U_{j_1}|$ and $\epsilon^{-2} \ge \epsilon^{-1}$, the bound (67) is the sum of the upper bounds (71) and (76). □

Remark 6 Theorem 3 also implies limit (47), which was established in Theorem 2. However, note that Theorem 2 was proved for $\epsilon_c > 0$, while to prove Theorem 3 we needed the stronger assumption $\epsilon_c = +\infty$.

Corollary 1 *Suppose that* A1–A5 *hold and let* $\epsilon \in (0,1]$. *Furthermore, assume that the model-improving algorithm requires at most* K *evaluations of* f *and* c *to construct fully linear models. Then, Algorithm* 1 *with rule* (48) *and* $\epsilon_c = +\infty$ *reaches an* ϵ*-critical point of* Φ *after at most*

$$O\left(K\left[|\log(\epsilon)| + |\log(\kappa_u)|\right]\epsilon^{-2}\right) \tag{77}$$

function evaluations (that is, evaluations of f *and* c*), where* $\kappa_u = (\kappa_s + \mu^{-1})\bar{\Delta}$.

Proof In the worst case, Algorithm 2 will be called in all iterations of Algorithm 1, and in each one of this calls, the number of repetitions in Algorithm 2 will be bounded above by

$$\left|\frac{\log\left(\frac{\epsilon}{\kappa_u}\right)}{\log(\omega)}\right|, \tag{78}$$

such that the ϵ-criticality criterion (52) in Lemma 13 is not satisfied. Since in each one of these repetitions the model-improving algorithm requires at most K evaluations of f and c to construct fully linear models, it follows that each iteration of Algorithm 1 requires at most

$$K\left|\frac{\log\left(\frac{\epsilon}{\kappa_u}\right)}{\log(\omega)}\right| \tag{79}$$

function evaluations. Hence, by Theorem 3, Algorithm 1 takes at most

$$K\left|\frac{\log\left(\frac{\epsilon}{\kappa_u}\right)}{\log(\omega)}\right|\kappa_w\epsilon^{-2} \tag{80}$$

function evaluations to reduce the criticality measure $\psi_1(x)$ below ϵ. □

Corollary 2 *Suppose that* A1–A5 *hold and let* $\epsilon \in (0, 1]$. *Furthermore, assume that Algorithm* 6.3 *in* Conn et al. (2009a) *is used as model-improving algorithm. Then, Algorithm* 1 *with rule* (48) *and* $\epsilon_c = +\infty$ *reaches an* ϵ*-critical point of* Φ *after at most*

$$O\left((n+1)\left[|\log(\epsilon)| + |\log(\kappa_u)|\right]\epsilon^{-2}\right) \tag{81}$$

function evaluations (*that is, evaluations of* f *and* c), *where* $\kappa_u = (\kappa_s + \mu^{-1})\bar{\Delta}$.

Proof It follows from the fact that Algorithm 6.3 in Conn et al. (2009a) takes at most $K = n + 1$ evaluations of f and c to construct fully linear models. □

To contextualize the results above, it is worth to review some recent works on worst-case complexity in nonconvex and nonlinear optimization. Regarding derivative-based algorithms, Cartis et al. (2011b) have proposed a first-order trust-region method[3] and a first-order quadratic regularization method for minimizing (1), for which they proved a worst-case complexity bound of $O\left(\epsilon^{-2}\right)$ function evaluations to reduce $\psi_1(x_k)$ below ϵ. On the other hand, in the context of derivative-free optimization, Cartis et al. (2012) have proved a worst-case complexity bound of

$$O\left((n^2 + 5n)\left[1 + |\log(\epsilon)|\right]\epsilon^{-\frac{3}{2}}\right) \tag{82}$$

function evaluations for their adaptive cubic regularization (ARC) algorithm applied to unconstrained smooth optimization, where derivatives are approximated by finite differences. Furthermore, for the minimization of a nonconvex and nonsmooth function, Nesterov

[3] Algorithm 1 with $H_k = 0$ for all k can be viewed as a derivative-free version of the first-order trust-region method proposed in Cartis et al. (2011b).

(2011) has proposed a random derivative-free smoothing algorithm for which he has proved a worst-case complexity bound of

$$O\left(n(n+4)^2\epsilon^{-3}\right) \tag{83}$$

function evaluations to reduce the expected squared norm of the gradient of the smoothing function below ϵ. Finally, still in the nonsmooth case, Garmanjani and Vicente (2013) have proposed a class of smoothing direct-search methods for which they proved a worst-case complexity bound of

$$O\left(n^{\frac{5}{2}}\left[|\log(\epsilon)| + \log(n)\right]\epsilon^{-3}\right) \tag{84}$$

function evaluations to reduce the norm of the gradient of the smoothing function below ϵ.

Comparing the worst-case complexity bound (81) with the bounds of (82) and (83), we see that (81) is better in terms of the powers of n and ϵ. However, in such comparison, we must take into account that these worst-case complexity bounds were obtained for different criticality measures.

6 A derivative-free exact penalty algorithm

Let us now consider the equality-constrained optimization problem (3), where the objective function $f : \mathbb{R}^n \to \mathbb{R}$ and the constraint function $c : \mathbb{R}^n \to \mathbb{R}^r$ are continuously differentiable. One way to solve such problem consists of solving the related unconstrained problem

$$\min_{x\in\mathbb{R}^n} \Phi(x,\sigma), \tag{85}$$

where

$$\Phi(x,\sigma) \equiv f(x) + \sigma\|c(x)\| \tag{86}$$

is an exact penalty function and $\|.\|$ is a polyhedral norm. More specifically, an exact penalty algorithm for problem (3) can be stated as follows.
Algorithm 3 (Exact Penalty Algorithm - see, e.g., Sun and Yuan 2006)

```
Step 0  Given $x_0 \in \mathbb{R}^n$, $\sigma_0 > 0$, $\lambda > 0$ and $\epsilon > 0$, set $k := 0$.
Step 1  Find a solution $x_{k+1}$ of problem (85) with $\sigma = \sigma_k$ and starting at $x_k$ (or at $x_0$).
Step 2  If $\|c(x_{k+1})\| \leq \epsilon$, stop. Otherwise, set $\sigma_{k+1} = \sigma_k + \lambda$, $k := k+1$ and go to Step 1.
```

As mentioned in Sect. 1, problem (85) is a particular case of problem (1). For this problem, the criticality measure $\psi_1(x)$ becomes

$$\psi_1(x) = f(x) + \sigma\|c(x)\| - \min_{\|s\|\leq 1}\left\{f(x) + \nabla f(x)^{\mathrm{T}}s + \sigma\|c(x) + J_c(x)s\|\right\} \equiv \Psi_\sigma(x), \tag{87}$$

and we have the following result.

Theorem 4 *Suppose that* A1, A2 *and* A4 *hold, and assume that* $\{f(x_k)\}$ *is bounded below. Then, for any* $\epsilon \in (0, 1]$, *Algorithm* 1 *applied to problem* (85) *will generate a point* $x(\sigma)$ *such that*

$$\Psi_\sigma(x(\sigma)) \leq \epsilon. \tag{88}$$

Proof It follows directly from Theorem 2 (or from Theorem 3 when (48) is used and $\epsilon_c = +\infty$). □

By the above theorem, we can solve approximately subproblem (85) in Step 1 of Algorithm 3 by applying Algorithm 1. The result is the following derivative-free exact penalty algorithm.

Algorithm 4 (Derivative-Free Exact Penalty Algorithm)

Step 0 Given $x_0 \in \mathbb{R}^n$, $\sigma_0 > 0$, $\lambda > 0$ and $\epsilon > 0$, set $k := 0$.

Step 1 Apply Algorithm 1 to solve subproblem (85) with $\sigma = \sigma_k$. Start at x_k (or at x_0) and stop at an approximate solution x_{k+1} for which

$$\Psi_{\sigma_k}(x_{k+1}) \leq \epsilon,$$

where $\psi_{\sigma_k}(x_{k+1})$ is defined in (87) with $\sigma = \sigma_k$ and $x = x_{k+1}$.

Step 2 If $\|c(x_{k+1})\| \leq \epsilon$, stop. Otherwise, set $\sigma_{k+1} = \sigma_k + \lambda$, $k := k + 1$ and go to Step 1.

Definition 4 Given $\epsilon \in (0, 1]$, a point $x \in \mathbb{R}^n$ is said to be a ϵ-stationary point of problem (3) if there exists y^* such that

$$\|\nabla f(x) + J_c(x)^{\mathrm{T}} y^*\| \leq \epsilon. \tag{89}$$

If, additionally, $\|c(x)\| \leq \epsilon$, then x is said to be a ϵ-KKT point of problem (3).

The next result, due to Cartis et al. (2011b), establishes the relation between ϵ-critical points of $\Phi(., \sigma)$ and ϵ-KKT points of (3).

Theorem 5 *Suppose that* A1 *holds and let* $\sigma > 0$. *Consider minimizing* $\Phi(., \sigma)$ *by some algorithm and obtaining an approximate solution* x *such that*

$$\Psi_\sigma(x) \leq \epsilon, \tag{90}$$

for a given tolerance $\epsilon > 0$ *(that is,* x *is a* ϵ*-critical point of* $\Phi(., \sigma)$*). Then, there exists* $y^*(\sigma)$ *such that*

$$\|\nabla f(x) + J_c(x)^{\mathrm{T}} y^*(\sigma)\| \leq \epsilon. \tag{91}$$

Additionally, if $\|c(x)\| \leq \epsilon$*, then* x *is a* ϵ*-KKT point of problem* (3).

Proof See Theorem 3.1 in Cartis et al. (2011b). □

The theorem below provides a worst-case complexity bound for Algorithm 4.

Theorem 6 *Suppose that* A1, A2 *and* A4 *hold. Assume that* $\{f(x_k)\}$ *is bounded below. If there exists* $\bar{\sigma} > 0$ *such that* $\sigma_k \leq \bar{\sigma}$ *for all* k*, then Algorithm* 4 *(where Algorithm* 1 *in Step* 1 *satisfies the assumptions in Corollary* 1*) terminates either with an* ϵ*-KKT point or with an infeasible* ϵ*-stationary point of* (3) *in at most*

$$\frac{\bar{\sigma} K}{\lambda} \left| \frac{log\left(\frac{\epsilon}{\bar{\kappa}_u}\right)}{log(\omega)} \right| \left\lceil \bar{\kappa}_w \epsilon^{-2} \right\rceil \tag{92}$$

function evaluations, where $\bar{\kappa}_u$ *and* $\bar{\kappa}_w$ *are positive constants dependent of* $\bar{\sigma}$*, but are independent of the problem dimensions* n *and* r.

Proof Using Corollary 1, it follows as in the proof of part (i) of Theorem 3.2 in Cartis et al. (2011b). □

For constrained nonlinear optimization with derivatives, Cartis et al. (2011b) have proposed a first-order exact penalty algorithm, for which they have proved a worst-case complexity bound of $O(\epsilon^{-2})$ function evaluations. Another complexity bound of the same order was obtained by Cartis et al. (2014) under weaker conditions for a first-order short-step homotopy algorithm. Furthermore, the same authors have proposed in Cartis et al. (2013) an short-step ARC algorithm, for which they have proved an improved worst-case complexity bound of $O\left(\epsilon^{-3/2}\right)$ function evaluations. However, despite the existence of several derivative-free algorithms for constrained optimization (see, e.g., Bueno et al. 2013; Conejo et al. 2013, 2014; Diniz-Ehrhardt et al. 2011; Martínez and Sobral 2013), to the best of our knowledge, (92) is the first complexity bound for constrained nonlinear optimization without derivatives.

7 Numerical experiments

To investigate the computational performance of the proposed algorithm, we have tested a MATLAB implementation of Algorithm 1, called "DFMS", on a set of 43 finite minimax problems from Lukšan and Vlček (2000) and Di Pillo et al. (1993). Recall that the finite minimax problem is a particular case of the composite NSO problem (1) where $f = 0$ and $h(c) = \max_{1\leq i\leq r} c_i$, namely

$$\min_{x\in\mathbb{R}^n} \Phi(x) \equiv \max_{1\leq i\leq r} c_i(x). \tag{93}$$

In the code DFMS, we approximate each function c_i by a linear interpolation model of the form (11). The interpolation points around x_k are chosen using Algorithm 6.2 in Conn et al. (2009a). As model-improvement algorithm, we employ Algorithm 6.3 in Conn et al. (2009a). Furthermore, as $h(c) = \max_{1\leq i\leq r} c_i$ is a polyhedral function, we consider $\|.\| = \|.\|_\infty$ and $H_k = 0$ for all k. In this way, subproblems (23), (25), (26) and (29) are reduced to linear programming problems, which are solved using the MATLAB function linprog. Finally, we use (28) to update Δ_k^*.

First, with the purpose of evaluating the ability of the code DFMS to obtain accurate solutions for the finite minimax problem, we applied this code on the test set with a maximum number of function evaluations to be 2550 and the parameters in Step 0 as $\bar{\Delta} = 50$, $\Delta_0^* = 1$, $\alpha_0 = 0$, $\alpha_1 = 0.25$, $\gamma_1 = 0.5$, $\gamma_2 = 2$, $\epsilon_c = 10^{-4}$, $\mu = 1$, $\beta = 0.75$ and $\omega = 0.5$. Within the budget of function evaluations, the code was terminated when any one of the criteria below was satisfied:

$$\Delta_k < 10^{-4} \quad \text{or} \quad \Phi(x_k) > 0.98\Phi(x_{k-10}), \qquad k > 10. \tag{94}$$

A problem was considered solved when the solution $\bar{x}$ obtained by DFMS was such that

$$\Phi\text{-Error} \equiv \frac{|\Phi(\bar{x}) - \Phi^*|}{\max\{1, |\Phi(\bar{x})|, |\Phi^*|\}} \leq 10^{-2}, \tag{95}$$

where Φ^* is the optimal function value provided by Lukšan and Vlček (2000) and Di Pillo et al. (1993).

Problems and results are reported in Table 1, where "n", "r" and "$n\Phi$" represent the number of variables, the number of subfunctions and the number of function evaluations, respectively. Moreover, an entry "F" indicates that the code stopped due to some error.

Table 1 Numerical results for finite minimax problems

Problem	n	r	Φ^*	$n\Phi$	Φ-error
01. CB2	2	3	1.9522E+00	55	6.5961E−04
02. WF	2	3	0.0000E+00	373	2.1934E−08
03. Spiral	2	2	0.0000E+00	121	7.7601E−02
04. EVD52	3	6	3.5997E+00	109	1.8400E−04
05. Rosen-Suzuki	4	4	−4.4000E+01	281	3.8277E−09
06. Polak 6	4	4	−4.4000E+01	431	1.4373E−09
07. PBC 3	3	42	4.2021E−03	88	3.4122E−05
08. Bard*	3	30	5.0816E−02	104	5.0416E−05
09. Kowalik–Osborne*	4	22	8.0844E−03	150	9.4040E−05
10. Davidon 2	4	40	1.1571E+02	261	4.2693E−04
11. OET5	4	42	2.6360E−03	316	8.2882E−05
12. OET6	4	42	2.0161E−03	150	7.6847E−05
13. Gamma	4	61	1.2042E−07	70	9.9999E−01
14. EXP	5	21	1.2237E−04	F	F
15. PBC 1	5	60	2.2340E−02	444	1.7848E−05
16. EVD61	6	102	3.4905E−02	280	4.8678E−02
17. Transformer	6	11	1.9729E−01	344	3.5529E−01
18. Filter	9	82	6.1853E−03	381	1.1538E−01
19. Wong 1	7	5	6.8063E+02	121	5.4263E−03
20. Wong 2	10	9	2.4306E+01	375	2.5243E−02
21. Wong 3	20	18	1.3373E+02	862	5.3498E−03
22. Polak 2	10	2	5.4598E+01	199	1.2651E−04
23. Polak 3	11	10	2.6108E+02	289	2.6148E−03
24. Watson*	20	62	1.4743E−08	294	9.9999E−01
25. Osborne 2	11	130	4.8027E−02	600	9.7441E−05
26. Crescent	2	2	0.0000E+00	132	5.9209E−08
27. CB3	2	3	2.0000E+00	79	2.7993E−03
28. DEM	2	3	−3.0000E+00	366	9.2632E−07
29. QL	2	3	7.2000E+00	49	4.4769E−06
30. LQ	2	2	−1.4142E+00	520	1.1270E−08
31. Shor	5	10	2.2600E+01	97	1.4379E−04
32. Maxquad	10	5	−8.4141E−01	155	8.4141E−01
33. Gill	10	3	9.7858E+00	551	1.0000E+00
34. Maxq	20	20	0.0000E+00	1,135	1.4524E−08
35. Maxl	20	40	0.0000E+00	504	1.5809E−13
36. MXHILB	50	100	0.0000E+00	F	F
37. Polak 1	2	2	2.7183E+00	301	3.0213E−06
38. Char.-Conn 1	2	3	1.9522E+00	52	5.3573E−04

Table 1 continued

Problem	n	r	Φ^*	$n\Phi$	Φ-error
39. Char.-Conn 2	2	3	1.9522E+00	79	2.7993E−03
40. Demy-Malo.	2	3	−3.0000E+00	366	9.2632E−07
41. Hald-Madsen 1	2	4	0.0000E+00	54	8.4185E−08
42. Hald-Madsen 2	5	42	1.2200E−04	F	F
43. El Attar	6	102	3.4900E−02	238	9.9953E−01

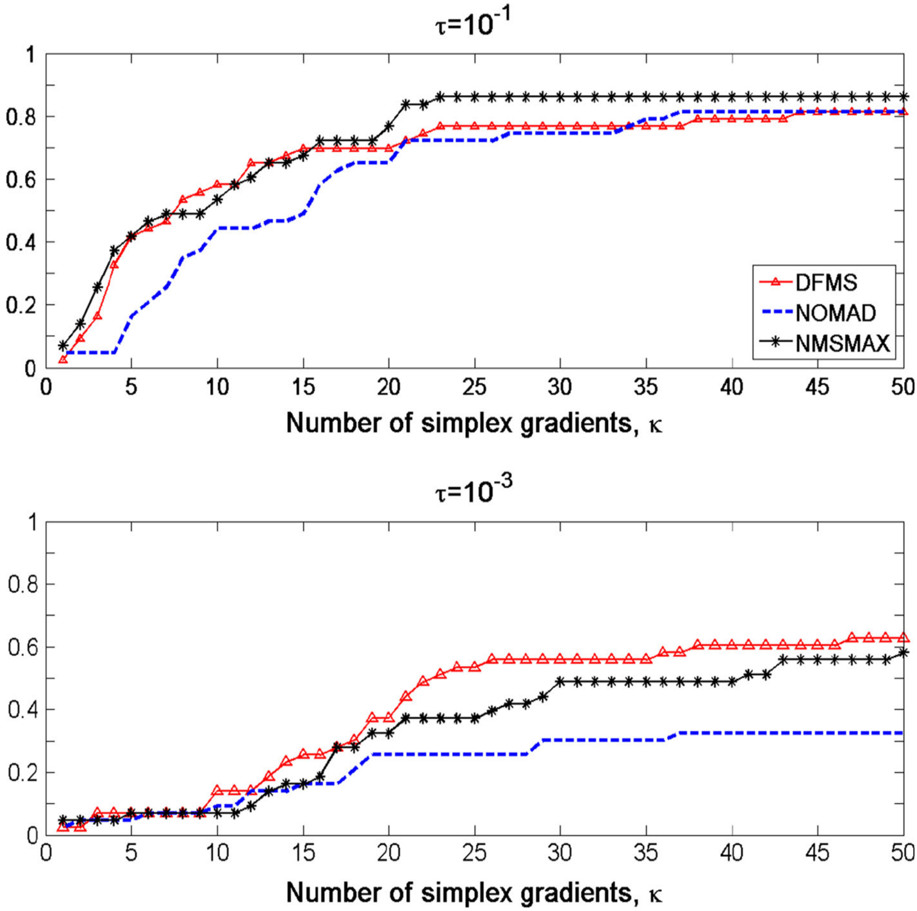


Fig. 1 Data profiles for the finite minimax problems show the percentage of problems solved as a function of a computational budget of simplex gradients. Here, we have the graphs for tolerances $\tau = 10^{-1}$ and $\tau = 10^{-3}$

The results given in Table 1 show that DFMS was able to solve most of the problems in the test set (30 of them), with a reasonable number of function evaluations. The exceptions were problems 3, 13, 16–18, 20, 24, 32, 33 and 43, where criterion (95) was not satisfied, and problems 14, 36 and 42, in which the code stopped due to some error.

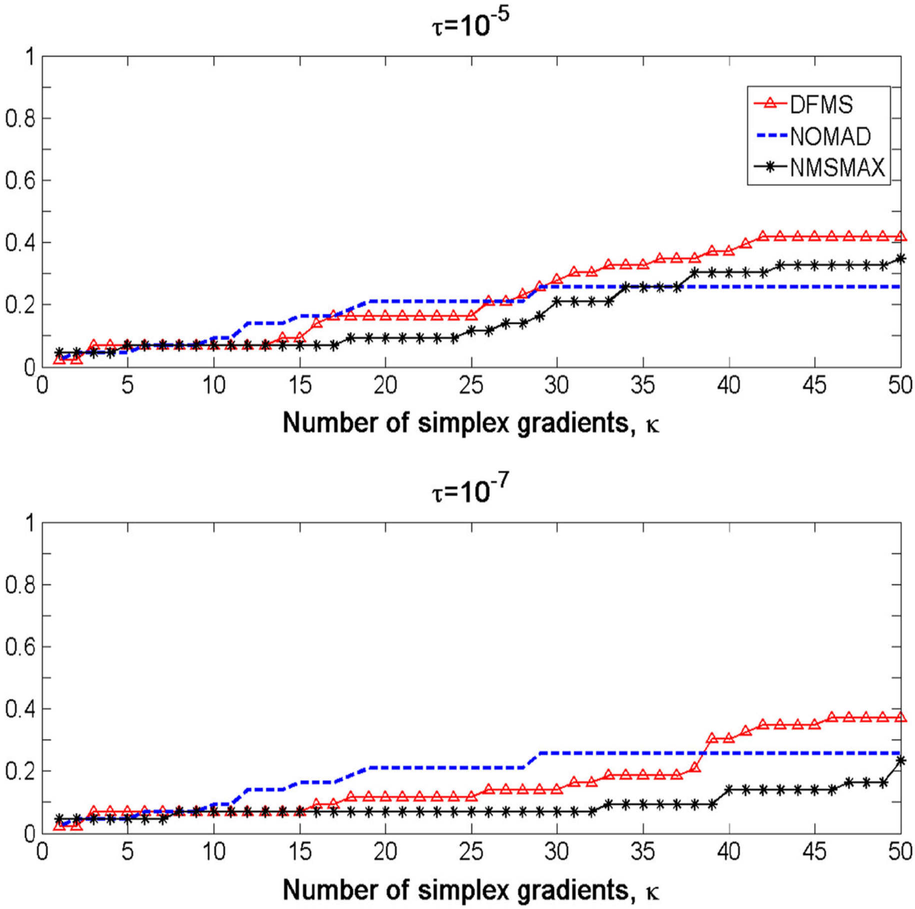


Fig. 2 Data profiles for the finite minimax problems show the percentage of problems solved as a function of a computational budget of simplex gradients. Here, we have the graphs for tolerances $\tau = 10^{-5}$ and $\tau = 10^{-7}$

To investigate the potentialities and limitations of Algorithm 1, we also compare DFMS with the following codes:

- NMSMAX: a MATLAB implementation of the Nelder–Mead method (Nelder and Mead 1965), freely available from the Matrix Computation Toolbox;[4] and
- NOMAD (version 3.6.2): a MATLAB implementation of the MADS algorithm (Le Digabel 2011), freely available from the OPTI Toolbox.[5]

We consider the data profile of Moré and Wild (2009). The convergence test for the codes is:

$$\Phi(x_0) - \Phi(x) \geq (1 - \tau)\left(\Phi(x_0) - \Phi_L\right), \tag{96}$$

where $\tau > 0$ is a tolerance, x_0 is the starting point for the problem, and Φ_L is computed for each problem as the smallest value of Φ obtained by any solver within a given number μ_Φ

[4] http://www.maths.manchester.ac.uk/~higham/mctoolbox.

[5] http://www.i2c2.aut.ac.nz/Wiki/OPTI/index.php/Main/HomePage.

of function evaluations. Since the problems in our test set have at most 50 variables, we set $\mu_{\Phi} = 2550$ so that all solvers can use at least 50 simplex gradients. The data profiles are presented for $\tau = 10^{-k}$ with $k \in \{1, 3, 5, 7\}$.

Despite the simplicity of our implementation of Algorithm 1, the data profiles in Figs. 1 and 2 suggest that DFMS is competitive with NMSMAX and NOMAD on finite minimax problems. This result is not so surprising, since DFMS exploits the structure of the finite minimax problem, which is not considered by codes NMSMAX and NOMAD.

8 Conclusions

In this paper, we have presented a derivative-free trust-region algorithm for composite NSO. The proposed algorithm is an adaptation of the derivative-free trust-region algorithm of Conn et al. (2009b) for unconstrained smooth optimization, with elements of the trust-region algorithms proposed by Fletcher (1982a), Powell (1983), Yuan (1985a) and Cartis et al. (2011b) for composite NSO. Under suitable conditions, weak global convergence was proved. Furthermore, considering a slightly modified update rule for the trust-region radius and assuming that the models are fully linear at all iterations, we adapted the argument of Cartis et al. (2011a, b) to prove a function-evaluation complexity bound for the algorithm reducing the criticality measure below ϵ. This complexity result was then specialized to the case when the composite function is an exact penalty function, providing a worst-case complexity bound for equality-constrained optimization problems, when the solution is computed using a derivative-free exact penalty algorithm. Finally, preliminary numerical experiments were done, and the results suggest that the proposed algorithm is competitive with the Nelder–Mead algorithm (Nelder and Mead 1965) and the MADS algorithm (Audet and Dennis 2006) on finite minimax problems.

Future research includes the conducting of extensive numerical tests using more sophisticated implementations of Algorithm 1. Further, it is worth to mention that the mechanism used in Algorithm 2 for ensuring the quality of the models is the simplest possible, but not necessarily the most efficient. Perhaps, the self-correcting scheme proposed by Scheinberg and Toint (2010) can be extended to solve composite NSO problems.

Acknowledgments This work was partially done while the G. N. Grapiglia was visiting the Institute of Computational Mathematics and Scientific/Engineering Computing of the Chinese Academy of Sciences. He would like to register his deep gratitude to Professor Ya-xiang Yuan, Professor Yu-hong Dai, Dr. Xin Liu and Dr. Ya-feng Liu for their warm hospitality. He also wants to thank Dr. Zaikun Zhang for valuable discussions. Finally, the authors are very grateful to the anonymous referees, whose comments have significantly improved the paper. G.N. Grapiglia was supported by CAPES, Brazil (Grant PGCI 12347/12-4). J. Yuan was partially supported by CAPES and CNPq, Brazil. Y. Yuan was partially supported by NSFC, China (Grant 11331012).

References

Audet C, Dennis JE Jr (2006) Mesh adaptive direct search algorithms for constrained optimization. SIAM J Optim 17:188–217

Bannert T (1994) A trust region algorithm for nonsmooth optimization. Math Program 67:247–264

Brent RP (1973) Algorithms for minimization without derivatives. Prentice-Hall, Englewood Cliffs

Bueno LF, Friedlander A, Martínez JM, Sobral FNC (2013) Inexact restoration method for derivative-free optimization with smooth constraints. SIAM J Optim 23:1189–1213

Cartis C, Gould NIM, Toint PhL (2011a) Adaptive cubic regularisation methods for unconstrained optimization. Part II: worst-case function—and derivative—evaluation complexity. Math Program 130:295–319

Cartis C, Gould NIM, Toint PhL (2011b) On the evaluation complexity of composite function minimization with applications to nonconvex nonlinear programming. SIAM J Optim 21:1721–1739

Cartis C, Gould NIM, Toint PhL (2012) On the oracle complexity of first-order and derivative-free algorithms for smooth nonconvex minimization. SIAM J Optim 22:66–86

Cartis C, Gould NIM, Toint PhL (2013) On the evaluation complexity of cubic regularization methods for potentially rank-deficient nonlinear least-squares problems and its relevance to constrained nonlinear optimization. SIAM J Optim 23:1553–1574

Cartis C, Gould NIM, Toint PhL (2014) On the complexity of finding first-order critical points in constrained nonlinear optimization. Math Program 144:93–106

Conn AR, Scheinberg K, Vicente LN (2009a) Introduction to derivative-free optimization. SIAM, Philadelphia

Conn AR, Scheinberg K, Vicente LN (2009b) Global convergence of general derivative-free trust-region algorithms to first and second order critical points. SIAM J Optim 20:387–415

Conejo PD, Karas EW, Pedroso LG, Ribeiro AA, Sachine M (2013) Global convergence of trust-region algorithms for convex constrained minimization without derivatives. Appl Math Comput 20:324–330

Conejo PD, Karas EW, Pedroso LG (2014) A trust-region derivative-free algorithm for constrained optimization. Technical Report. Department of Mathematics, Federal University of Paraná

Di Pillo G, Grippo L, Lucidi S (1993) A smooth method for the finite minimax problem. Math Program 60:187–214

Diniz-Ehrhardt MA, Martínez JM, Pedroso LG (2011) Derivative-free methods for nonlinear programming with general lower-level constraints. Comput Appl Math 30:19–52

Fletcher R (1982a) A model algorithm for composite nondifferentiable optimization problems. Math Program 17:67–76

Fletcher R (1982b) Second order correction for nondifferentiable optimization. In: Watson GA (ed) Numerical analysis. Springer, Berlin, pp 85–114

Garmanjani R, Vicente LN (2013) Smoothing and worst-case complexity for direct-search methods in nonsmooth optimization. IMA J Numer Anal 33:1008–1028

Grapiglia GN, Yuan J, Yuan Y (2014) On the convergence and worst-case complexity of trust-region and regularization methods for unconstrained optimization. Math Prog. doi:10.1007/s10107-014-0794-9

Greenstadt J (1972) A quasi-Newton method with no derivatives. Math Comput 26:145–166

Hooke R, Jeeves TA (1961) Direct search solution of numerical and statistical problems. J Assoc Comput Mach 8:212–229

Kelly JE (1960) The cutting plane method for solving convex programs. J SIAM 8:703–712

Le Digabel S (2011) Algorithm 909: NOMAD: Nonlinear optimization with the MADS algorithm. ACM Trans Math Softw 37:41:1–44:15

Lemaréchal C (1978) Bundle methods in nonsmooth optimization. In: Lemaréchal C, Mifflin R (eds) Nonsmooth optimization. Pergamon, Oxford, pp 79–102

Luksăn L, Vlček J (2000) Test problems for nonsmooth unconstrained and linearly constrained optimization. Technical Report 198, Academy of Sciences of the Czech Republic

Madsen K (1975) Minimax solution of non-linear equations without calculating derivatives. Math Program Study 3:110–126

Martínez JM, Sobral FNC (2013) Derivative-free constrained optimization in thin domains. J Glob Optim 56:1217–1232

Mifflin R (1975) A superlinearly convergent algorithm for minimization without evaluating derivatives. Math Program 9:100–117

Moré JJ, Wild SM (2009) Benchmarking derivative-free optimization algorithms. SIAM J Optim 20:172–191

Nelder JA, Mead R (1965) A simplex method for function minimization. Comput J 7:308–313

Nesterov Y (2011) Random gradient-free minimization of convex functions. Technical Report 2011/1, CORE

Powell MJD (1983) General algorithm for discrete nonlinear approximation calculations. In: Chui CK, Schumaker LL, Ward LD (eds) Approximation theory IV. Academic Press, New York, pp 187–218

Powell MJD (1984) On the global convergence of trust region algorithms for unconstrained minimization. Math Program 29:297–303

Powell MJD (2006) The NEWUOA software for unconstrained optimization without derivatives. In: Di Pillo G, Roma M (eds) Large nonlinear optimization. Springer, New York, pp 255–297

Qi L, Sun J (1994) A trust region algorithm for minimization of locally Lipschitzian functions. Math Program 66:25–43

Scheinberg K, Toint PhL (2010) Self-correcting geometry in model-based algorithms for derivative-free unconstrained optimization. SIAM J Optim 20:3512–3532

Shor NZ (1978) Subgradient methods: a survey of the Soviet research. In: Lemaréchal C, Mifflin R (eds) Nonsmooth optimization. Pergamon, Oxford

Stewart GW (1967) A modification of Davidon's minimization method to accept difference approximations of derivatives. J Assoc Comput Mach 14:72–83

Sun W, Yuan Y (2006) Optimization theory and methods: nonlinear programming. Springer, Berlin

Winfield D (1973) Function minimization by interpolation in a data table. J Inst Math Appl 12:339–347

Yuan Y (1985a) Conditions for convergence of trust region algorithms for nonsmooth optimization. Math Program 31:220–228

Yuan Y (1985b) On the superlinear convergence of a trust region algorithm for nonsmooth optimization. Math Program 31:269–285