SIAM J. OPTIM.
Vol. 28, No. 2, pp. 1367–1395

ON REGULARIZATION AND ACTIVE-SET METHODS WITH COMPLEXITY FOR CONSTRAINED OPTIMIZATION*

E. G. BIRGIN† AND J. M. MARTÍNEZ‡

Abstract. The main objective of this research is to introduce a practical method for smooth bound-constrained optimization that possesses worst-case evaluation complexity $O(\varepsilon^{-3/2})$ for finding an ε-approximate first-order stationary point when the Hessian of the objective function is Lipschitz continuous. As other well-established algorithms for optimization with box constraints, the algorithm proceeds visiting the different faces of the domain aiming to reduce the norm of an internal projected gradient and abandoning active constraints when no additional progress is expected in the current face. The introduced method emerges as a particular case of a method for minimization with linear constraints. Moreover, the linearly constrained minimization algorithm is an instance of a minimization algorithm with general constraints whose implementation may be unaffordable when the constraints are complicated. As a procedure for leaving faces, a different method is employed that may be regarded as an independent device for constrained optimization. Such an independent algorithm may be employed to solve linearly constrained optimization problems on its own, without relying on the active-set strategy. A careful implementation and numerical experiments show that the algorithm that combines active sets with leaving-face iterations is more effective than the independent algorithm on which leaving-face iterations are based, although both exhibit similar complexities $O(\varepsilon^{-3/2})$.

Key words. nonlinear programming, bound-constrained minimization, active-set strategies, regularization, complexity

AMS subject classifications. 90C30, 65K05, 49M37, 90C60, 68Q25

DOI. 10.1137/17M1127107

1. Introduction.

In this paper, we address the problem of minimizing a smooth and generally nonconvex function within a region of the Euclidean finite-dimensional space. Initially, we present a cubic regularization algorithm with cubic descent that finds an approximate first-order stationary point with arbitrary precision ε or a boundary point of the feasible region with evaluation complexity $O(\varepsilon^{-3/2})$. In addition, complexity for finding second-order stationary points and convergence results are presented. Secondly, we introduce an algorithm for minimization on arbitrary and generally nonconvex regions defined by inequalities and equalities. The evaluation complexity of this algorithm for finding first-order stationary points is also $O(\varepsilon^{-3/2})$ when one uses cubic regularization of the functional quadratic approximation. The most general form, in which we use pth Taylor polynomials to define subproblems, has complexity $O(\varepsilon^{-(p+1)/p})$. A version of this algorithm for minimizing smooth functions with linear constraints is introduced and implemented for the case $p = 2$. Moreover, the problem of minimizing with linear constraints is also addressed in a different way. Namely, we consider each face of the polytope as a finite-dimensional region within which we may employ the algorithm first introduced in this paper. As mentioned

*Received by the editors April 25, 2017; accepted for publication (in revised form) January 26, 2018; published electronically May 8, 2018.
http://www.siam.org/journals/siopt/28-2/M112710.html
Funding: Supported by FAPESP (grants 2013/03447-6, 2013/05475-7, 2013/07375-0, and 2014/18711-3) and CNPq (grants 309517/2014-1 and 303750/2014-6).
†Dept. of Computer Science, Institute of Mathematics and Statistics, University of São Paulo, Rua do Matão, 1010, Cidade Universitária, 05508-090, São Paulo, SP, Brazil (egbirgin@ime.usp.br).
‡Dept. of Applied Mathematics, Institute of Mathematics, Statistics, and Scientific Computing, State University of Campinas, 13083-859, Campinas, SP, Brazil (martinez@ime.unicamp.br).

above, such an algorithm either finds an approximate stationary point within the face or finds a point on its boundary. Then, as a natural combination of the first two algorithms so far introduced, we use the first one for minimizing within faces and the second one for giving up constraints when the current face is exhausted. This gives rise to a third algorithm with complexity $O(\varepsilon^{-3/2})$ for finding ε-approximate first-order stationary points. This method is implemented and compared with the second one. Therefore, the present paper introduces an algorithm with complexity $O(\varepsilon^{-3/2})$ for finding approximate KKT points of linearly constrained optimization problems, as a result of the manipulation of a theoretical algorithm for constrained optimization. Theoretical algorithms for finding approximate KKT points of general nonlinear programming problems with complexity $O(\varepsilon^{-3/2})$ have already been introduced [9], but practical counterparts are not yet known.

Several numerical optimization traditions converge on the present work. Algorithms that use quadratic models for unconstrained problems and employ regularized or trust-regions subproblems combined with actual-versus-predicted reduction or cubic descent with proven complexity $O(\varepsilon^{-3/2})$ were given in [15, 20, 21, 22, 31, 39, 47, 51]. Cubic regularization was introduced as a useful optimization tool in [45, 53]. The optimality of the complexity $O(\varepsilon^{-3/2})$ was proved in [20]. The generalization of cubic regularization to arbitrary $(p+1)$th regularization was given in [10]. In [44], the complexity (close to $O(\varepsilon^{-2})$) of quasi-Newton methods for unconstrained optimization was analyzed. The complexity of unconstrained optimization and some constrained optimization algorithms assuming relaxed Lipschitz (Hölder) conditions on the objective function was analyzed in [27, 43, 46]. The idea of minimizing on polytopes using a fast algorithm within faces combined with a suitable procedure to abandon exhausted faces is in the core of active-set methods for constrained optimization and may be found in several textbooks [14, 40, 52]. Many algorithms for solving general constrained optimization problems use subproblems that consist of the minimization of combinations of objective and constraint functions subject to linear constraints [14, 40, 50, 52]. The combined algorithm presented here uses ideas of [3, 4, 12, 13], where the algorithm for leaving faces is the spectral projected gradient method [1, 16, 17, 18, 19]. In fact, Algorithm 4.1 presented in this work is based on the active-set strategies [3, 4, 12, 13] that, for leaving faces, use, as well as Algorithm 4.1, the comparison of different components of the projected gradient as in [14, 33, 35, 41]. On the other hand, in Algorithm 2.1 (which is used within the faces in Algorithm 4.1), successive iterates are computed using safeguarded regularization whereas in [3] the trust-region approach is employed and [4, 12, 13] use a line search approach. Moreover, Algorithm 4.1 introduced in this work and the algorithms introduced in [3, 4, 12, 13] also differ in the way of reaching the boundary and in the fact that the first one possesses a complexity analysis.

This paper is organized as follows. Section 2 introduces the cubic regularization method that either finds an interior stationary point or stops at the boundary of the feasible region. First- and second-order complexity results are presented. In section 3, we introduce a model algorithm with $(p+1)$-regularized models for nonlinear programming. In section 4, we describe the method for minimization with linear constraints that combines the procedures of sections 2 and 3. In section 5, we compare the combined algorithm with the method presented in section 3 with $p = 2$ and with the method introduced in [3]. In section 6, we state some conclusions and lines for future research.

Notation. Given $\Omega \subseteq \mathbb{R}^n$, $\mathrm{Int}(\Omega)$ denotes the set of interior points of Ω and $\bar{\Omega}$ denotes its closure; if Ω is convex, $P_\Omega(x)$ denotes the Euclidean projection of x onto Ω; ∇ and ∇^2 are the gradient and Hessian operators, respectively; $\|\cdot\|$ denotes the Euclidean norm; $\mathbb{N} = \{0, 1, 2, \dots\}$; $g(x)$ denotes the gradient of $f(x)$ and $H(x)$ denotes its Hessian; $\lambda_1(H)$ denotes the leftmost eigenvalue of the symmetric matrix H; H_k denotes $H(x^k)$; $h'(x)$ denotes the Jacobian of $h : \mathbb{R}^n \to \mathbb{R}^q$; if $a \in \mathbb{R}$, $[a]_+ = \max\{a, 0\}$; and $A^\dagger$ denotes the Moore–Penrose pseudoinverse of a matrix A.

2. Inner-to-the-face algorithm.

Consider the problem

$$\text{Minimize } f(x) \text{ subject to } x \in \Omega.$$

We assume that $\Omega \subset \mathbb{R}^n$ has nonempty interior and $f : \mathbb{R}^n \to \mathbb{R}$. In this section, we introduce an algorithm that finds an interior point that approximately satisfies first- and second-order conditions for unconstrained minimization; or, alternatively, it finds a point on the boundary of Ω. More precisely, either the algorithm generates a *finite* sequence $x^0, x^1, \dots, x^{\hat{k}}$ such that the final point $x^{\hat{k}}$ is on the boundary of Ω, $x^k \in \mathrm{Int}(\Omega)$ for all $k < \hat{k}$, and $f(x^{\hat{k}}) < f(x^{\hat{k}-1}) < \cdots < f(x^0)$; or it generates an *infinite* sequence $\{x^k\}_{k=0}^\infty \subset \mathrm{Int}(\Omega)$ such that $\{f(x^k)\}$ is strictly decreasing, $g(x^k) \to 0$, and $[-\lambda_1(H(x^k))]_+ \to 0$ when $k \to \infty$.

In the algorithm, for a given iterate x^k and a trial step s^{trial}, we consider the interiority condition

$$x^k + s^{\mathrm{trial}} \in \mathrm{Int}(\Omega) \tag{1}$$

and the sufficient descent condition

$$f(x^k + s^{\mathrm{trial}}) \le f(x^k) - \alpha \|s^{\mathrm{trial}}\|^3. \tag{2}$$

The first step of the algorithm consists in computing, when possible, a solution s^{trial} to the quadratic model

$$\text{Minimize } g(x^k)^T s + \frac{1}{2} s^T H_k s \tag{3}$$

and checking whether it satisfies (1), (2) or not. Step 2 (which is executed when the first step is not successful) consists in, by increasing the regularizing parameter ρ in a controlled way, finding a solution $s^{\mathrm{trial}} = s^{\mathrm{trial}}(\rho)$ to the cubic regularized model

$$\text{Minimize } g(x^k)^T s + \frac{1}{2} s^T H_k s + \rho \|s\|^3 \tag{4}$$

that satisfies (1), (2). The algorithm computes exact solutions to (4) with a regularization parameter ρ that is not known a priori but belongs to a specified interval. Exact solutions are affordable in this case and using them is crucial for the practical performance of the algorithm. Step 3 consists in verifying whether the regularizing parameter ρ happened to be too large or not. Completing the verification may require the calculation of a point on the boundary of Ω and this task is performed at Step 4. At the end, the computed new iterate may be a solution to (3) or (4) that belongs to the interior of Ω and satisfies the sufficient descent condition or a point on the boundary of Ω. In the latter case, the algorithm stops. The complete description of the algorithm follows.

ALGORITHM 2.1. *Assume that* $x^0 \in \mathrm{Int}(\Omega)$, $\alpha > 0$, $\tau_2 \geq \tau_1 > 1 > \tau_0 > 0$, $M > 0$, $\hat{\rho}_0^{\mathrm{ini}} \in (0, M/\tau_1]$, $\nu \in \{0, 1\}$, *and* $J \geq 0$ *are given. Initialize* $k \leftarrow 0$.

Step 1. *Set* $j \leftarrow 0$.

Step 1.1. *If* (3) *is not solvable, go to Step* 2.

Step 1.2. *Define* $\rho_{k,j} = 0$ *and* $s^{k,j}$ *as a solution to* (3).

Step 1.3. *If* (1) *does not hold with* $s^{\mathrm{trial}} = s^{k,j}$ *and* $j < J$, *compute, when possible, a point* w *on the boundary of* Ω. *If* $f(w) < f(x^k)$ *then define* $x^{k+1} = w$ *and stop.*

Step 1.4. *If* (1) *and* (2) *hold with* $s^{\mathrm{trial}} = s^{k,j}$, *define* $s^k = s^{k,j}$, $\rho_k = \rho_{k,j}$, $x^{k+1} = x^k + s^k$, *and* $\hat{\rho}_{k+1}^{\mathrm{ini}} = \nu\ \tau_0 \hat{\rho}_k^{\mathrm{ini}} + (1 - \nu)\ \hat{\rho}_k^{\mathrm{ini}}$, *set* $k \leftarrow k + 1$ *and go to Step* 1. *Otherwise, set* $j \leftarrow j + 1$ *(and continue at Step* 2 *below).*

Step 2. *Set* $\hat{\rho} \leftarrow \hat{\rho}_k^{\mathrm{ini}}$.

Step 2.1. *Define* $s^{k,j}$ *as a solution to* (4) *with* $\rho = \rho_{k,j}$ *for some* $\rho_{k,j} \in [\tau_1 \hat{\rho}, \tau_2 \hat{\rho}]$.

Step 2.2. *If* (1) *does not hold with* $s^{\mathrm{trial}} = s^{k,j}$ *and* $j < J$, *compute, when possible, a point* w *on the boundary of* Ω. *If* $f(w) < f(x^k)$ *then define* $x^{k+1} = w$ *and stop.*

Step 2.3. *If* (1) *or* (2) *does not hold with* $s^{\mathrm{trial}} = s^{k,j}$, *set* $\hat{\rho} \leftarrow \rho_{k,j}$, $j \leftarrow j + 1$ *and go to Step* 2.1.

Step 3. *If*

$$\rho_{k,j} \leq M \quad \text{or} \quad j = 0 \quad \text{or} \quad x^k + s^{k,j-1} \in \mathrm{Int}(\Omega), \tag{5}$$

define $s^k = s^{k,j}$, $\rho_k = \rho_{k,j}$, $x^{k+1} = x^k + s^k$, *and* $\hat{\rho}_{k+1}^{\mathrm{ini}} = \nu\ \tau_0 \rho_{k,j} + (1 - \nu)\ \hat{\rho}_k^{\mathrm{ini}}$, *set* $k \leftarrow k + 1$ *and go to Step* 1.

Step 4. *Compute* $w = x^k + s$ *on the boundary of* Ω *in such a way that* s *is a solution to* (4) *with* $\rho \in [\rho_{k,j-1}, \rho_{k,j})$. *(If* $x^k + s^{k,j-1}$ *is on the boundary of* Ω *then* $w = x^k + s^{k,j-1}$ *is the natural choice.)*

Step 4.1. *If* $f(w) < f(x^k)$ *then define* $x^{k+1} = w$ *and stop. Otherwise, define* $s^k = s^{k,j}$, $\rho_k = \rho_{k,j}$, $x^{k+1} = x^k + s^k$, *and* $\hat{\rho}_{k+1}^{\mathrm{ini}} = \nu\ \tau_0 \rho_{k,j} + (1 - \nu)\ \hat{\rho}_k^{\mathrm{ini}}$, *set* $k \leftarrow k + 1$ *and go to Step* 1.

Algorithm 2.1 has been described in such a way that the only reason for stopping is to find an iterate on the boundary of Ω. In any other case, the algorithm generates an infinite sequence. Of course, in practice, stopping criteria are necessary (and they will be defined later) but, in theory, we are interested on the behavior of the potentially infinite sequence of iterates.

At Step 1.1, (3) is solvable if and only if H_k is positive definite or H_k is positive semidefinite and $g(x^k)$ belongs to the range space of H_k. Note also that, at Step 2.1, we solve the cubic regularization problem (4) with a regularization parameter ρ which is a priori unknown and may take any value between $\tau_1 \hat{\rho}$ and $\tau_2 \hat{\rho}$. This may be done using a root-finding process that aims to compute the quadratic regularization parameter that corresponds to a cubic regularization one between the given bounds (see [15] for details). Recall that, as shown in [21, Thm. 3.1], the set of solutions to cubic regularized problems, varying ρ, coincides with the set of solutions to quadratic regularized problems, varying σ.

At Steps 1.3 and 2.2, there exists the possibility of computing a magical step $w - x^k$ whose definition may depend on characteristics of Ω. Since, in practice, magical steps may rely on computing projections, we have in mind the case in which projecting onto Ω is an affordable task, like it is, for example, in boxes, balls, and spheres. This

means that, if (1) does not hold with $s^{\text{trial}} = s^{k,j}$, we may define w as the projection of $x^k + s^{k,j}$ onto Ω. This trick has been proved to be very useful in practice. A similar device is used in [49]. The number of magical steps per iteration is limited to J.

When the algorithm arrives to Step 3, a point $x^k + s^{k,j}$ such that (1) and (2) hold with $s^{\text{trial}} = s^{k,j}$ has been computed. If, in addition, $\rho_{k,j} \leq M$, $x^k + s^{k,j}$ is accepted as the new iterate. The obvious question is why we do not accept the trial point $x^k + s^{k,j}$, in spite of it being interior and satisfying the sufficient descent condition, when (5) does not hold. The reason is that, since $\rho_{k,j} > M$ could be very big, $s^{k,j}$ could be unacceptably small and, so, sufficient descent could not mean satisfactory descent. This situation may only happen when $j > 0$ and $x^k + s^{k,j-1} \notin \text{Int}(\Omega)$, which means that the previous trial point at the present iteration was rejected without testing its functional value because it was not interior. Accepting $x^k + s^{k,j}$ as a new iterate or not involves computing an additional point at the boundary of Ω and this task is performed at Step 4.

The explanation that follows, related to the computation of a point on the boundary of Ω at Step 4, requires some background on trust-region and regularization subproblems, which may be found, for example, in [29, 15, 21, 47]. Step 4 starts with $x_{\text{int}} = x^k + s_{\text{int}} \in \text{Int}(\Omega)$, which satisfies (2), and $x_{\text{out}} = x^k + s_{\text{out}} \notin \text{Int}(\Omega)$, where $s_{\text{int}} = s^{k,j}$ and $s_{\text{out}} = s^{k,j-1}$ are solutions to (4) with $\rho = \rho_{\text{int}} = \rho_{k,j}$ and $\rho = \rho_{\text{out}} = \rho_{k,j-1}$, respectively, and $\rho_{\text{int}} > \rho_{\text{out}} \geq 0$. This means that both s_{int} and s_{out} come from solving quadratic regularization problems with regularization parameters $\sigma_{\text{int}} = 3\|s_{\text{int}}\|\rho_{\text{int}}$ and $\sigma_{\text{out}} = 3\|s_{\text{out}}\|\rho_{\text{out}}$, respectively, with $\sigma_{\text{int}} \geq \sigma_{\text{out}} \geq 0$ (see [47, Lem. 2.3]). If x_{out} is in the border of Ω then, at Step 4, we may define $w = x_{\text{out}}$ and there is nothing else to be done. Therefore, from now on we assume that $x_{\text{out}} \notin \Omega$. If $\sigma_{\text{int}} = \sigma_{\text{out}}$, we are in the so-called hard-case of the trust-region literature [29]. In this case, all the points in the segment $[s_{\text{int}}, s_{\text{out}}]$ are solutions to (4) for values of ρ between ρ_{int} and ρ_{out}. Therefore, there is a point s_{bound} on the segment $[s_{\text{int}}, s_{\text{out}}]$ such that s_{bound} solves (4) for some ρ between ρ_{out} and ρ_{int} and $w = x^k + s_{\text{bound}}$ is on the boundary of Ω. Such s_{bound} may be obtained by means of bisection on the segment $[s_{\text{int}}, s_{\text{out}}]$ or analytically computed if the constraints are simple enough. If $\sigma_{\text{int}} > \sigma_{\text{out}}$ and s_{out} is of the form $s_{\text{out}} = -(H_k + \sigma_{\text{out}} I)^{\dagger} g(x^k)$, we necessarily have that $s_{\text{int}} = -(H_k + \sigma_{\text{int}} I)^{-1} g(x^k)$ and the required point on the boundary may be found by bisection on the interval $[\sigma_{\text{out}}, \sigma_{\text{int}}]$. Finally, it may be that $\sigma_{\text{int}} > \sigma_{\text{out}}$ and s_{out} is of the form $s_{\text{out}} = -(H_k + \sigma_{\text{out}} I)^{\dagger} g(x^k) + tv$ with $t > 0$, where v is an eigenvector of H_k associated with its leftmost eigenvalue. This is also a hard-case situation. We handle it testing whether the point $x^k - (H_k + \sigma_{\text{out}} I)^{\dagger} g(x^k)$ is interior to Ω or not. If it is interior, the required point on the boundary may be found by bisection on the segment $[-(H_k + \sigma_{\text{out}} I)^{\dagger} g(x^k), s_{\text{out}}]$ or analytically computed if the constraints are simple enough. Otherwise, the boundary point may be obtained by bisection on the interval $[\sigma_{\text{out}}, \sigma_{\text{int}}]$.

Remark 2.1. An attentive reader may observe that, when we rely on a bisection-like process, the required point w on the boundary is obtained only in the limit, and not in a finite number of steps. This minor problem may be solved as follows. Assume that $[x_{\text{int}}, x_{\text{out}}]$ is the segment whose extremes are computed by the bisection procedure and that $\|x_{\text{int}} - x_{\text{out}}\| \leq \tau_{\text{small}} \|x_{\text{int}} - x^k\|^3$. At Step 4, we compute $w \in [x_{\text{int}}, x_{\text{out}}]$ such that w is on the boundary of Ω. Note that w may be analytically computed if the constraints are simple enough. In Remark 2.2, which follows Lemma 2.2, we show that this modification does not affect the theoretical proofs.

Assumption A1. There exists $L > 0$ such that for all x^k computed by Algorithm 2.1 and all s^{trial} considered at (2) we have that

$$f(x^k + s^{\text{trial}}) \leq f(x^k) + g(x^k)^T s^{\text{trial}} + \frac{1}{2}(s^{\text{trial}})^T H_k s^{\text{trial}} + L\|s^{\text{trial}}\|^3. \tag{6}$$

LEMMA 2.1. *Suppose that Assumption* A1 *holds. If* $\rho_{k,j} \geq L + \alpha$ *then* (2) *holds with* $s^{\text{trial}} = s^{k,j}$.

Proof. If $\rho_{k,j} \geq L + \alpha$, by Assumption A1, we have that

$$\begin{aligned} f(x^k + s^{k,j}) &\leq f(x^k) + g(x^k)^T s^{k,j} + \tfrac{1}{2}(s^{k,j})^T H_k s^{k,j} + L\|s^{k,j}\|^3 \\ &= f(x^k) + g(x^k)^T s^{k,j} + \tfrac{1}{2}(s^{k,j})^T H_k s^{k,j} + (L+\alpha)\|s^{k,j}\|^3 - \alpha\|s^{k,j}\|^3 \\ &\leq f(x^k) + g(x^k)^T s^{k,j} + \tfrac{1}{2}(s^{k,j})^T H_k s^{k,j} + \rho_{k,j}\|s^{k,j}\|^3 - \alpha\|s^{k,j}\|^3. \end{aligned}$$

Since $s^{k,j}$ being a solution to (4) with $\rho = \rho_{k,j}$ implies that

$$g(x^k)^T s^{k,j} + \frac{1}{2}(s^{k,j})^T H_k s^{k,j} + \rho_{k,j}\|s^{k,j}\|^3 \leq 0,$$

(2) follows from the last inequality. ∎

LEMMA 2.2. *Suppose that Assumption* A1 *holds. Then, the sequence* $\{x^k\}$ *generated by Algorithm* 2.1 *is well defined. Moreover, if the algorithm does not stop at a boundary point at iteration* k *(in which case* ρ_k *is undefined), we have that*

$$\rho_k \leq \max\{M, \tau_2(L + \alpha)\}. \tag{7}$$

Proof. Proving that the sequence $\{x^k\}$ is well defined consists in showing that, given $x^k \in \text{Int}(\Omega)$, the point x^{k+1} is computed in finite time. If x^{k+1} is a solution to (3) computed at Step 1 or it is a point in the boundary of Ω computed at Steps 1.3, 2.2, or 4, we are done since those calculations involve a finite number of operations by definition. We now assume that $x^{k+1} = x^k + s^{k,j}$ and $s^{k,j}$ is a solution to (4) with $\rho = \rho_{k,j}$ (computed at Step 3). Since x^k is interior, for $\rho_{k,j}$ large enough we have that $x^k + s^{k,j}$ is interior too, meaning that (1) holds with $s^{\text{trial}} = s^{k,j}$. Moreover, by Lemma 2.1, if $\rho_{k,j} \geq L + \alpha$ then (2) holds with $s^{\text{trial}} = s^{k,j}$. Thus, since the updating rule of $\rho_{k,j}$ implies that $\rho_{k,j} \to \infty$ when $j \to \infty$, we have that $x^k + s^{k,j}$ satisfying (1), (2) with $s^{\text{trial}} = s^{k,j}$ is found in finite time.

Let us now prove the boundedness of ρ_k. If $\rho_k = \rho_{k,j}$ is defined at Step 3 then we have that $\rho_k \leq M$. Assume now that $\rho_k = \rho_{k,j}$ is defined at Step 4.1. By the definition of the algorithm, we have that $\rho_{k,j} \in [\tau_1\rho_{k,j-1}, \tau_2\rho_{k,j-1}]$. Moreover, the point w on the boundary of Ω was rejected. This point is of the form $w = x^k + s_w$ with s_w being a solution to (4) with ρ_w satisfying $\rho_{k,j-1} \leq \rho_w < \rho_{k,j}$. The point w was rejected because $f(w) \geq f(x^k)$, which means that (2) with $s^{\text{trial}} = s_w$ does not hold. Therefore, by Lemma 2.1, we have that $\rho_w \not\geq L + \alpha$. Thus, $\rho_{k,j-1} < L + \alpha$ and, in consequence, $\rho_{k,j} < \tau_2(L + \alpha)$ as we wanted to prove. ∎

Remark 2.2. Assume that the point w on the boundary is computed as described in Remark 2.1. Let $\gamma > 0$ be a Lipschitz constant for the objective function f. Then, in the case that $f(w) \geq f(x^k)$, we have that

$$\|w - x_{\text{int}}\| \leq \|x_{\text{out}} - x_{\text{int}}\| \leq \tau_{\text{small}}\|x_{\text{int}} - x^k\|^3.$$

Then,

$$f(x_{\rm int}) \geq f(x^k) - \tau_{\rm small}\gamma\|x_{\rm int} - x^k\|^3.$$

This means that $x_{\rm int}$ is a solution to the cubic regularized subproblem that does not satisfy the sufficient descent condition related to $\alpha = \tau_{\rm small}\gamma$. Then, by Lemma 2.1, $x_{\rm int}$ comes from a regularization parameter ρ smaller than $L + \tau_{\rm small}\gamma$. Then, as in Lemma 2.2, we would have $\rho_k \leq \tau_2(L + \tau_{\rm small}\gamma)$. This is enough to complete the complexity proofs.

The proofs of the next two lemmas are standard and may be essentially found, for example, in [24] and [51].

LEMMA 2.3. *If $s^k = 0$ then $g(x^k) = 0$ and H_k is positive semidefinite.*

Proof. Since

$$\nabla\left[g(x^k)^T s + \frac{1}{2}s^T H_k s\right]\Big|_{s=0} = g(x^k)$$

and $\nabla^2[g(x^k)^T s + \frac{1}{2}s^T H_k s]|_{s=0} = H_k$, if $s^k = 0$ solves (3) then we have that $g(x^k) = 0$ and H_k is positive semidefinite.

The gradient of the objective function that defines (4) is $g(x^k) + H_k s + 3\rho\|s\|s$. Therefore, if $s^k = 0$ is a solution to (4) then we have that $g(x^k) = 0$. Moreover, for all $s \in \mathbb{R}^n$, we must have

$$\frac{1}{2}s^T H_k s + \rho\|s\|^3 \geq 0,$$

otherwise $s^k = 0$ would not be a solution to (4). Then, for all $s \neq 0$,

$$\frac{1}{2}\frac{s^T H_k s}{\|s\|^2} + \rho\|s\| \geq 0.$$

In particular, if $s \neq 0$ is an eigenvector associated with $\lambda_1(H_k)$, it turns out that

$$\frac{1}{2}\lambda_1(H_k) + \rho\|s\| \geq 0.$$

Taking limits for $s \to 0$ it follows that $\lambda_1(H_k) \geq 0$ as we wanted to prove. ∎

LEMMA 2.4. *Suppose that Assumption* A1 *holds, that Algorithm* 2.1 *generates an infinite sequence $\{x^k\}_{k=0}^{\infty}$, and that $\{f(x^k)\}_{k=0}^{\infty}$ is bounded below. Then,*

$$\lim_{k\to\infty} \|s^k\| = 0 \tag{8}$$

and

$$\lim_{k\to\infty} [-\lambda_1(H_k)]_+ = 0. \tag{9}$$

Proof. Since $\{f(x^k)\}$ is bounded below, (8) follows from the fulfillment of the sufficient descent condition (2) with $s^{\rm trial} = s^k$ for all k.

Moreover, s^k is a minimizer of $g(x^k)s + \frac{1}{2}s^T H_k s + \rho_k\|s\|^3$ and, by Lemma 2.3, H_k is positive semidefinite if $s^k = 0$. If $s^k \neq 0$, the Hessian of the cubic function $g(x^k)s + \frac{1}{2}s^T H_k s + \rho_k\|s\|^3$ must be positive semidefinite. This Hessian is

$$H_k + 3\rho_k\left(\frac{s^k(s^k)^T}{\|s^k\|} + \|s^k\|I\right).$$

Thus,

$$\left[-\lambda_1\left(H_k + 3\rho_k \frac{s^k (s^k)^T}{\|s^k\|} + \|s^k\| I\right)\right]_+ \geq 0.$$

Since $s^k \to 0$ and, by Lemma 2.2, ρ_k is bounded, we have that (9) holds. ∎

Assumption A2. Assumption A1 holds and, for all x^k and s^k computed by Algorithm 2.1,

$$\|g(x^k + s^k)\| \leq (L + 3\rho_k)\|s^k\|^2. \tag{10}$$

A sufficient condition for the fulfillment of Assumption A2 comes from assuming that $g(x^k + s^k)$ is well represented by its linear approximation, namely,

$$\|g(x^k + s^k) - g(x^k) - H(x^k)s^k\| \leq L\|s^k\|^2. \tag{11}$$

In this case, Assumption A2 holds due to the gradient annihilation of the subproblem at Step 2. Moreover, a sufficient condition for the fulfillment of both Assumptions A1 and A2 is the fulfillment of a Lipschitz condition by the Hessian $H(x)$. It is interesting to note that Assumption A2 requires (10) to be verified only with respect to increments s^k that satisfy the interiority and the sufficient descent conditions, whereas Assumption A1 requires (6) to hold at every trial increment s^{trial}. This remark is related to the weak-Lipschitz condition and the concept of path of iterates in [23, 25, 30].

LEMMA 2.5. *Suppose that Assumption* A2 *holds. Then, for all x^k and s^k generated by Algorithm* 2.1 *such that $x^{k+1} = x^k + s^k \in \text{Int}(\Omega)$, we have that*

$$f(x^{k+1}) \leq f(x^k) - \alpha\left(\frac{\|g(x^{k+1})\|}{L + 3\max\{M, \tau_2(L+\alpha)\}}\right)^{3/2}. \tag{12}$$

Proof. By Assumption A2 and Lemma 2.2, we have that

$$\|g(x^{k+1})\| \leq (L + 3\max\{M, \tau_2(L+\alpha)\})\|s^k\|^2. \tag{13}$$

Thus, (12) follows from (13) and the fact that (2) holds with $s^{\text{trial}} = s^k$. ∎

THEOREM 2.1. *Suppose that Assumption* A2 *holds and let $f_{\text{target}} \in \mathbb{R}$, $\varepsilon_g > 0$, and $\varepsilon_{\text{H}} > 0$ be arbitrary. Let $\rho_{\max} = \max\{M, \tau_2(L+\alpha)\}$. Then, the number of iterations at which*

$$f(x^k) > f_{\text{target}} \quad \textit{and} \quad \|g(x^k)\| > \varepsilon_g$$

is, at most,

$$\kappa_1 = \left\lfloor \left(\frac{f(x^0) - f_{\text{target}}}{\alpha}\right)\left(\frac{\varepsilon_g}{L + 3\rho_{\max}}\right)^{-3/2} \right\rfloor \tag{14}$$

and the number of iterations at which

$$f(x^k) > f_{\text{target}} \quad \textit{and} \quad \lambda_1(H_k) < -\varepsilon_{\text{H}}$$

is, at most,

$$\kappa_2 = \left\lfloor \left(\frac{f(x^0) - f_{\text{target}}}{\alpha}\right)\left(\frac{\varepsilon_{\text{H}}}{6\rho_{\max}}\right)^{-3} \right\rfloor. \tag{15}$$

Moreover, when $\nu = 0$, the number of functional evaluations per iteration is bounded by

$$J + \left\lfloor \log_{\tau_1} \left(\frac{\rho_{\max}}{\hat{\rho}_0^{\text{ini}}} \right) \right\rfloor + 2 \tag{16}$$

and, when $\nu = 1$, the total number of functional evaluations performed at iterations at which

$$f(x^k) > f_{\text{target}} \quad \textit{and} \quad \|g(x^k)\| > \varepsilon_g$$

is, at most,

$$\left\lfloor \log_{\tau_1} \left(\frac{\rho_{\max}}{\hat{\rho}_0^{\text{ini}}} \right) + \left| \log_{\tau_1}(\tau_0) \right| \kappa_1 \right\rfloor + (J+2)(\kappa_1 + 1) \tag{17}$$

and the total number of functional evaluations performed at iterations at which

$$f(x^k) > f_{\text{target}} \quad \textit{and} \quad \lambda_1(H_k) < -\varepsilon_{\text{H}}$$

is, at most,

$$\left\lfloor \log_{\tau_1} \left(\frac{\rho_{\max}}{\hat{\rho}_0^{\text{ini}}} \right) + \left| \log_{\tau_1}(\tau_0) \right| \kappa_2 \right\rfloor + (J+2)(\kappa_2 + 1). \tag{18}$$

Proof. The maximum number of iterations (14) follows directly from Lemma 2.5.

The step s^k is a minimizer of $g(x^k)^T s + \frac{1}{2} s^T H_k s + \rho_k \|s\|^3$, where, by Lemma 2.2, $\rho_k \leq \rho_{\max}$. Thus, if $s^k \neq 0$, the Hessian of $g(x^k)^T s + \frac{1}{2} s^T H_k s + \rho_k \|s\|^3$ is positive semidefinite at $s = s^k$. By direct calculations, we see that this Hessian is given by

$$H_k + 3\rho_k \left[\frac{s^k (s^k)^T}{\|s^k\|} + \|s^k\| I \right].$$

Let v^k be an eigenvector of H_k associated with $\lambda_1(H_k)$ and $\|v^k\| = 1$. Then, by the positive definiteness of $H_k + 3\rho[\frac{s^k (s^k)^T}{\|s^k\|} + \|s^k\| I]$, we have that

$$\begin{aligned} 0 &\leq v_k^T \left(H_k + 3\rho_k \left[\tfrac{s^k (s^k)^T}{\|s^k\|} + \|s^k\| I \right] \right) v^k \\ &\leq \lambda_1(H_k) + 3\rho_k \left[(v_k^T s_k)^2 / \|s^k\| + \|s^k\| \right] \\ &\leq \lambda_1(H_k) + 3\rho_k \left[\|s^k\|^2 / \|s^k\| + \|s^k\| \right] \\ &\leq \lambda_1(H_k) + 6\rho_{\max} \|s^k\|. \end{aligned}$$

Thus, $\|s^k\| \geq -\lambda_1(H_k)/(6\rho_{\max})$ or, equivalently, $-\|s^k\|^3 \leq (\lambda_1(H_k)/(6\rho_{\max}))^3$. Thus, since (2) with $s^{\text{trial}} = s^k$ holds for all k, we have that

$$f(x^{k+1}) \leq f(x^k) - \alpha \|s^k\|^3 \leq f(x^k) + \alpha \left([\lambda_1(H_k)]/(6\rho_{\max}) \right)^3,$$

from which (15) follows.

Regarding the number of functional evaluations, let us consider first the case $\nu = 0$. In order to establish the number of functional evaluations per iteration k, first note that, while the interiority condition (1) is not being satisfied with $s^{\text{trial}} = s^{k,j}$, on the one hand there is no need to check whether the descent condition (2) holds with

$s^{\text{trial}} = s^{k,j}$ and, on the other hand, simple decrease may be checked at no more than J magical steps. This means that, at every iteration k, while (1) does not hold with $s^{\text{trial}} = s^{k,j}$, at most J functional evaluations are performed. Once (1) is satisfied, the limit (16) on the number of functional evaluations follows from the boundedness of ρ_k for all k, given by Lemma 2.2, the fact that $\rho_{k,0} = 0$ by definition, and the updating rules for $\rho_{k,j}$ that guarantee that (a) if (3) is solvable then $\rho_{k,j} \geq 2^{j-1}\hat{\rho}_k^{\text{ini}} = 2^{j-1}\hat{\rho}_0^{\text{ini}}$ for all $j \geq 1$ and (b) if (3) is not solvable then $\rho_{k,j} \geq 2^j \hat{\rho}_k^{\text{ini}} = 2^j \hat{\rho}_0^{\text{ini}}$ for all $j \geq 0$.

Let us consider now the case in which $\nu = 1$. Let j_k be such that $\rho_k = \rho_{k,j_k}$. By the definition of the algorithm, the number of functional evaluations at iteration k is smaller than or equal to $J + j_k + 2$. In order to establish a limit on the total number of functional evaluations, we will construct a bound for $\sum_{\ell=0}^{k} j_\ell$ for any $k \geq 0$.

By the definition of the algorithm, for all $k \geq 0$, we have that (a) if $j_k = 0$ then $\hat{\rho}_{k+1}^{\text{ini}} = \tau_0 \hat{\rho}_k^{\text{ini}}$; (b) if $j_k > 0$ and (3) is not solvable then $\hat{\rho}_{k+1}^{\text{ini}} = \tau_0 \rho_{k,j_k} \geq \tau_0 \tau_1^{j_k+1} \hat{\rho}_k^{\text{ini}}$; and (c) if $j_k > 0$ and (3) is solvable then $\hat{\rho}_{k+1}^{\text{ini}} = \tau_0 \rho_{k,j_k} \geq \tau_0 \tau_1^{j_k} \hat{\rho}_k^{\text{ini}}$. This means that, for all $k \geq 0$, $\hat{\rho}_{k+1}^{\text{ini}} \geq \tau_0 \tau_1^{j_k} \hat{\rho}_k^{\text{ini}}$. Therefore, an inductive argument shows that, for all $k \geq 0$,

$$\hat{\rho}_{k+1}^{\text{ini}} \geq \tau_0^k \tau_1^{(\sum_{\ell=0}^{k} j_\ell)} \hat{\rho}_0^{\text{ini}}. \tag{19}$$

On the other hand, since, for all $k \geq 0$, by Lemma 2.2, $\rho_{k,j_k} = \rho_k \leq \rho_{\max}$, we have that

$$\hat{\rho}_{k+1}^{\text{ini}} \leq \tau_0 \rho_{\max} \leq \rho_{\max}. \tag{20}$$

Therefore, by (19) and (20), for all $k \geq 0$, we have that

$$\sum_{\ell=0}^{k} j_\ell \leq \left\lfloor \log_{\tau_1} \left(\frac{\rho_{\max}}{\hat{\rho}_0^{\text{ini}}} \right) + \left| \log_{\tau_1}(\tau_0) \right| k \right\rfloor.$$

Thus, for all $k \geq 0$,

$$\sum_{\ell=0}^{k} (J + j_\ell + 2) \leq \left\lfloor \log_{\tau_1} \left(\frac{\rho_{\max}}{\hat{\rho}_0^{\text{ini}}} \right) + \left| \log_{\tau_1}(\tau_0) \right| k \right\rfloor + (J+2)(k+1),$$

and, replacing k with κ_1 in (14) and with κ_2 in (15), we obtain the desired results (17) and (18), respectively. This completes the proof. ∎

THEOREM 2.2. *Suppose that Assumption* A2 *holds and that the sequence* $\{f(x^k)\}$ *generated by Algorithm* 2.1 *is bounded. Then, either the sequence stops at some boundary point* $x^{\hat{k}}$ *such that* $f(x^{\hat{k}}) < f(x^{\hat{k}-1}) < \cdots < f(x^0)$ *or an infinite sequence is generated such that*

$$\lim_{k\to\infty} \|g(x^k)\| = 0 \quad \textit{and} \quad \lim_{k\to\infty} [-\lambda_1(H_k)]_+ = 0.$$

Proof. If the sequence stops at a boundary point $x^{\hat{k}}$ then $f(x^{\hat{k}}) < f(x^{\hat{k}-1}) < \cdots < f(x^0)$ follows by the definition of the algorithm. Assume that the sequence does not stop at a boundary point. Since the sequence is bounded, by continuity, there exists $f_{\text{bound}} \in \mathbb{R}$ such that $f(x^k) > f_{\text{bound}}$ for all k. Define $f_{\text{target}} = f_{\text{bound}}$. Let $\varepsilon > 0$ be arbitrary. By Theorem 2.1, taking $\varepsilon_g = \varepsilon$ we see that there exists k_0 such that, for all $k \geq k_0$, $\|g(x^k)\| \leq \varepsilon$. The fact that $\lim_{k\to\infty}[-\lambda_1(H_k)]_+ = 0$ has been proved in Lemma 2.4. This completes the proof. ∎

3. High-order algorithms for constrained optimization. Consider the problem

$$\text{Minimize } f(x) \text{ subject to } x \in D, \tag{21}$$

where $D \subset \mathbb{R}^n$ represents an arbitrary set. In this section, we introduce a class of methods for solving (21). The algorithms to be introduced in this section can be used in connection with the algorithm introduced in section 2 in different ways. Therefore, this class of algorithms may be seen as independent procedures for solving the main problem (21) or as auxiliary devices for continuing Algorithm 2.1 when "a face" of D should be abandoned.

Algorithm 3.1 below is a high-order algorithm in which each iteration is defined by the approximate minimization of the pth Taylor approximation of the function f around the iterate x^k as in [10]. In [10], only unconstrained problems are considered and acceptance of trial points is based on the comparison of actual and predicted reductions, whereas here we use cubic descent with safeguards. If $f : \mathbb{R}^n \to \mathbb{R}$ with continuous derivatives up to order $p \in \{1, 2, 3, \dots\}$, the Taylor polynomial of order p may be written in the form

$$T_p(x, s) = \sum_{j=1}^{p} P_j(x, s), \tag{22}$$

where $P_j(x, s)$ is an homogeneous polynomial of degree j given by

$$P_j(x, s) = \frac{1}{j!} \left(s_1 \frac{\partial}{\partial x_1} + \cdots + s_n \frac{\partial}{\partial x_n} \right)^j f(x). \tag{23}$$

For example, $T_1(x, s) = g(x)^T s$ and $T_2(x, s) = g(x)^T s + \frac{1}{2} s^T H(x) s$.

ALGORITHM 3.1. *Assume that $p \in \{1, 2, 3, \dots\}$, $\alpha > 0$, $\rho_{\min} > 0$, $\beta > 0$, $\varepsilon_d > 0$, $\tau_2 \geq \tau_1 > 1$, $\theta > 0$, and $x^0 \in D$ are given. Initialize $k \leftarrow 0$.*

Step 1. *Set $\rho \leftarrow 0$.*

Step 2. *Compute $s^{\text{trial}} \in \mathbb{R}^n$ such that*

$$x^k + s^{\text{trial}} \in D \tag{24}$$

and

$$T_p(x^k, s^{\text{trial}}) + \rho \|s^{\text{trial}}\|^{p+1} \leq 0. \tag{25}$$

Step 3. *Test the conditions*

$$f(x^k + s^{\text{trial}}) \leq f(x^k) - \alpha \|s^{\text{trial}}\|^{p+1} \tag{26}$$

and

$$f(x^k + s^{\text{trial}}) \leq f(x^k) - \beta \varepsilon_d^{(p+1)/p}. \tag{27}$$

If (26) holds, accept the step s^{trial}. If only (27) holds, accept s^{trial} or not according to criteria that will be specified later. If s^{trial} is accepted, define $s^k = s^{\text{trial}}$, $\rho_k = \rho$, and $x^{k+1} = x^k + s^k$, set $k \leftarrow k + 1$ and go to Step 1. Otherwise, update $\rho \leftarrow \max\{\rho_{\min}, \tau\rho\}$ with $\tau \in [\tau_1, \tau_2]$ and go to Step 2.

Algorithm 3.1 is analogous to the main algorithm of [46]. In [46], only Hölder conditions on the pth derivatives are used, instead of the Lipschitz conditions employed in Algorithm 3.1. Very likely both Algorithms 2.1 and 3.1, as well as the combined Algorithm 4.1 (which will be presented in the next section), may be adapted, relaxing Lipschitz continuity to Hölder continuity. However, the adaptation involves technical difficulties that are beyond the scope of the present paper. Condition (27) is not necessary for proving any of the theoretical results of Algorithm 3.1. This condition has been used in [6] and [7]. As will be seen later, it is an option to (26) that may be useful in cases in which an approximate solution to the subproblem (28) below is computationally hard to obtain.

The trial increment s^{trial} is intended to be an approximate solution to the subproblem

$$\text{Minimize } T_p(x^k, s) + \rho\|s\|^{p+1} \text{ subject to } x^k + s \in D. \tag{28}$$

Differently from Algorithm 2.1, in Algorithm 3.1, we do not compute exact solutions to the regularized problem (28). This is because, due to the presence of constraints in this subproblem, exact solutions are not easily available. The condition (25) is the minimal condition that should be imposed on an approximate solution to (28) in order to obtain meaningful results, although an obvious choice that satisfies (24) and (25) is $s^{\text{trial}} = 0$, which is not useful at all and will be discarded later. On the other hand, note that (24) imposes full-precision feasibility on the approximate solutions s^{trial} to (28) and, thus, it imposes some conditions on the kind of feasible sets D that can be tackled in practice.

LEMMA 3.1. *Assume that the sequence $\{x^k\}$ is generated by Algorithm* 3.1. *If $\{f(x^k)\}$ is bounded below then*

$$\lim_{k\to\infty} \|s^k\| = 0.$$

Proof. If f is bounded below, the number of iterations at which $s^k = s^{\text{trial}}$ satisfies (27) is obviously finite. Therefore, after a finite number of iterations, all the accepted iterates satisfy (26). The proof follows straightforwardly from the hypotheses of the lemma and (26). ∎

The following assumption coincides with Assumption A1 in the case in which $p = 2$.

Assumption A1. There exists $L > 0$ such that for all x^k computed by Algorithm 3.1 and all s^{trial} considered at (26) we have that

$$f(x^k + s^{\text{trial}}) \le f(x^k) + T_p(x^k, s^{\text{trial}}) + L\|s^{\text{trial}}\|^{p+1}.$$

LEMMA 3.2. *Suppose that Assumption* A1 *holds. If the regularization parameter ρ in* (25) *satisfies $\rho \ge L + \alpha$ then a trial step s^{trial} that satisfies* (25) *also satisfies the sufficient descent condition* (26).

Proof. By Assumption A1,

$$\begin{aligned} f(x^k + s^{\text{trial}}) &\le f(x^k) + T_p(x^k, s^{\text{trial}}) + L\|s^{\text{trial}}\|^{p+1} \\ &= f(x^k) + T_p(x^k, s^{\text{trial}}) + \rho\|s^{\text{trial}}\|^{p+1} - \rho\|s^{\text{trial}}\|^{p+1} + L\|s^{\text{trial}}\|^{p+1}. \end{aligned}$$

Then, by (25),

$$f(x^k + s^{\text{trial}}) \le f(x^k) - \rho\|s^{\text{trial}}\|^{p+1} + L\|s^{\text{trial}}\|^{p+1}.$$

Therefore, if $\rho \ge L + \alpha$, (26) holds. ∎

Assumption A2. Assumption A1 holds and, for all x^k and s^k computed by Algorithm 3.1 such that $s^k = s^{\text{trial}}$ satisfies (26), we have that

$$\|g(x^k + s^k) - \nabla_s T_p(x^k, s^k)\| \leq L\|s^k\|^p. \tag{29}$$

Assumptions A1 and A2 are satisfied if the pth derivatives of f satisfy a Lipschitz condition (see [10]). As in the case of Assumptions A1 and A2, observe that Assumption A1 must hold for every trial increment s^{trial}, whereas Assumption A2 must hold only at the accepted increments s^k.

Assumption A3. The set D is defined by

$$D = \{x \in \mathbb{R}^n \mid h_i(x) \leq 0 \text{ for all } i = 1, \ldots, q\}, \tag{30}$$

where the functions h_i are continuously differentiable. Moreover, every feasible point of (30) satisfies a constraint qualification.

Assumption A3 implies that, for any function $\varphi : \mathbb{R}^n \to \mathbb{R}$, if z is a minimizer of φ subject to $z \in D$, associated KKT conditions hold. For the sake of simplicity and without loss of generality, we formulated the definition of D only in terms of inequality constraints. Implicitly, we assume that if equality constraints are present, they are expressed as pairs of inequalities.

For all $x \in D$, we define

$$L(D, x) = \{z \in \mathbb{R}^n \mid h_i(x) + h_i'(x)(z - x) \leq 0 \text{ for all } i = 1, \ldots, q\}.$$

We say that $L(D, x)$ is the linear approximation of D around x. Given $x \in D$ and a function φ, we define $\nabla_D \varphi(x) = P_{L(D,x)}(x - \nabla\varphi(x)) - x$. This notion has been introduced by Dunn [36, 37, 38] and used several times in the optimization literature. See, for example, [28]. Direct calculations show that x satisfies the KKT conditions of the problem of minimizing φ onto D if and only if $\nabla_D \varphi(x) = 0$. If there exists $x^k \to x^*$ such that $\nabla_D \varphi(x^k) \to 0$, we say that x^* satisfies the approximate gradient projection (AGP) sequential optimality condition [2, 48]. A worst-case complexity analysis for a constrained optimization algorithm that uses ∇_D may be found in [26].

Assumption A4. There exists $\theta > 0$ such that, for all $k \in \mathbb{N}$, s^k satisfies

$$f(x^k + s^k) \leq f(x^k) - \beta\varepsilon_d^{(p+1)/p} \tag{31}$$

or

$$\left\| \nabla_D \left[T_p(x^k, x - x^k) + \rho\|x - x^k\|^{p+1} \right] \big|_{x = x^k + s^k} \right\| \leq \theta\|s^k\|^p. \tag{32}$$

Assumption A4 with (32) states that the accepted increment s^k satisfies, approximately, an AGP optimality condition. If the constraints satisfy some constraint qualification, every minimizer of (28) satisfies the AGP condition and, consequently, also fulfills (32). Therefore, (32) states the degree of accuracy with which one wishes to solve the subproblems. Note that Assumption A4 eliminates the possibility of taking $s^k = s^{\text{trial}} = 0$. Note also that the degree of accuracy (32) is required not for all the subproblems but only to the ones that, ultimately, define algorithmic progress. Condition (31) (or (27)) is an alternative to the combination of (26) plus (32) for the case in which, in practice, the algorithm that solves the subproblems (28) has difficulties in satisfying (32).

LEMMA 3.3. *Suppose that Assumptions* A2*,* A3*, and* A4 *hold and that the sequence* $\{x^k\}$ *is generated by Algorithm* 3.1*. Then, at each iteration* k *such that* $s^k = s^{\text{trial}}$ *satisfies* (26) *and* (32)*, we have that*

$$\left\|\nabla_D f(x^k + s^k)\right\| \leq (L + \tau_2 (L + \alpha)(p + 1) + \theta) \|s^k\|^p. \tag{33}$$

Proof. By the definition of ∇_D, we have that

$$\begin{aligned}
&\|\nabla_D f(x^k + s^k) - \nabla_D T_p(x^k, x - x^k)|_{x=x^k+s^k}\| \\
&= \|P_{L(D,x^k+s^k)}\left[(x^k + s^k) - g(x^k + s^k)\right] - \\
&\qquad P_{L(D,x^k+s^k)}\left[(x^k + s^k) - \nabla T_p(x^k, x - x^k)|_{x=x^k+s^k}\right]\| \qquad (34)\\
&\leq \|(x^k + s^k) - g(x^k + s^k) - \left((x^k + s^k) - \nabla T_p(x^k, x - x^k)|_{x=x^k+s^k}\right)\| \\
&= \|g(x^k + s^k) - \nabla T_p(x^k, x - x^k)|_{x=x^k+s^k}\| \leq L\|s^k\|^p,
\end{aligned}$$

where the first inequality follows from the contraction property of projections and the last inequality follows from Assumption A2. This means that

$$\begin{aligned}
&\|\nabla_D f(x^k + s^k)\| \\
&\leq L\|s^k\|^p + \|\nabla_D T_p(x^k, x - x^k)|_{x=x^k+s^k}\| \\
&\leq (L + \theta)\|s^k\|^p + \rho\|\nabla_D \left[\|x - x^k\|^{p+1}\right]|_{x=x^k+s^k}\| \\
&= (L + \theta)\|s^k\|^p + \rho\|P_{L(D,x^k+s^k)}\left[(x^k + s^k) - \nabla\left[\|x - x^k\|^{p+1}\right]|_{x=x^k+s^k}\right] - (x^k + s^k)\| \\
&\leq (L + \theta)\|s^k\|^p + \rho\|(x^k + s^k) - \nabla\left[\|x - x^k\|^{p+1}\right]|_{x=x^k+s^k} - (x^k + s^k)\| \\
&= (L + \theta)\|s^k\|^p + \rho\|\nabla\left[\|x - x^k\|^{p+1}\right]|_{x=x^k+s^k}\| \\
&= (L + \theta + \rho(p + 1))\|s^k\|^p,
\end{aligned}$$

where the first inequality follows from (34), the second inequality follows from (32), and the third inequality follows from the fact that $x^k + s^k$ belongs to $L(D, x^k + s^k)$ and, therefore, for any $z \in \mathbb{R}^n$, $P_{L(D,x^k+s^k)}(z)$ is closer to $x^k + s^k$ than z itself. Finally, by Lemma 3.2, we have that $\rho \leq \tau_2(L + \alpha)$, which implies the desired result. ∎

LEMMA 3.4. *Suppose that Assumptions* A2*,* A3*, and* A4 *hold and that the sequence* $\{x^k\}$ *is generated by Algorithm* 3.1*. Then, at each iteration* k *such that* $s^k = s^{\text{trial}}$ *satisfies* (26) *and* (32)*, we have that*

$$f(x^{k+1}) \leq f(x^k) - \alpha\left(\frac{\|\nabla_D f(x^{k+1})\|}{L + \tau_2 (L + \alpha)(p + 1) + \theta}\right)^{(p+1)/p}. \tag{35}$$

Proof. The result follows straightforwardly from Lemma 3.3 and (26). ∎

THEOREM 3.1. *Suppose that Assumptions* A2*,* A3*, and* A4 *hold and that the sequence* $\{x^k\}$ *is generated by Algorithm* 3.1*. Let* $f_{\text{target}} \in \mathbb{R}$ *and* $\varepsilon_d, \varepsilon_g > 0$ *be arbitrary. Then, the number of iterations* k *such that* $s^k = s^{\text{trial}}$ *satisfies* (26) *and* (32) *and*

$$f(x^k) > f_{\text{target}} \quad \textit{and} \quad \|\nabla_D f(x^{k+1})\| \geq \varepsilon_g$$

is not greater than

$$\left\lfloor \left(f(x^0) - f_{\text{target}}\right)\left(\frac{\alpha^{p/(p+1)}\varepsilon_g}{L + \tau_2 (L + \alpha)(p + 1) + \theta}\right)^{-(p+1)/p} \right\rfloor \tag{36}$$

and the number of iterations k such that (31) *holds and $f(x^k) > f_{\text{target}}$ is not greater than*

$$\left\lfloor \left(f(x^0) - f_{\text{target}}\right) \left(\beta^{p/(p+1)}\varepsilon_d\right)^{-(p+1)/p} \right\rfloor. \tag{37}$$

Moreover, the number of functional evaluations per iteration is bounded above by

$$\left\lfloor \log_{\tau_1}\left(\frac{\tau_2(L+\alpha)}{\rho_{\min}}\right)\right\rfloor + 1.$$

Finally, if $\{f(x^k)\}$ is bounded below, the number of iterations at which (31) *holds is finite and*

$$\lim_{k\to\infty} \|\nabla_D f(x^{k+1})\| = 0. \tag{38}$$

Proof. The bounds (36) and (37) on the number of iterations follow from Lemma 3.4 and (31), respectively. The bound on the number of functional evaluations per iteration follows from the updating rule for ρ and Lemma 3.2, while (38) follows from Lemma 3.4 and the boundedness of $\{f(x^k)\}$. ∎

Assumption A5. For all $k \in \mathbb{N}$, s^k is a global minimizer of $T_p(x^k, s) + \rho_k\|s\|^{p+1}$ subject to $x^k + s \in D$.

If the constraints $h(x) \le 0$ satisfy a constraint qualification, the global minimizers to the subproblem satisfy the KKT conditions and, so, Assumption A5 implies (32) in Assumption A4.

THEOREM 3.2. *Suppose that Assumption* A2*,* A3*,* A4*, and* A5 *hold and that the sequence $\{x^k\}$ is generated by Algorithm* 3.1*. Let x^* be a limit point of $\{x^k\}$. Then, there exists $\rho_* \in [0, \tau_2(L+\alpha)]$ such that $s = 0$ is a global minimizer of $T_p(x^*, s) + \rho_*\|s\|^{p+1}$ subject to $x^* + s \in D$.*

Proof. By Lemma 3.2, $\rho_k \in [0, \tau_2(L+\alpha)]$ for all $k \in \mathbb{N}$. Therefore, there exists $\rho_* \in [0, \tau_2(L+\alpha)]$ and an infinite sequence of indices K such that

$$\lim_{k\in K} \rho_k = \rho_* \quad \text{and} \quad \lim_{k\in K} x^k = x^*.$$

Let $s \in \mathbb{R}^n$ be arbitrary and such that $x^k + s \in D$. By Assumption A5, we have that

$$T_p(x^k, s^k) + \rho_k\|s^k\|^{p+1} \le T_p(x^k, s) + \rho_k\|s\|^{p+1}.$$

Taking limits in this inequality for $k \in K$ and using the continuity of the derivatives of f up to order p, the fact that, by Lemma 3.1, $s^k \to 0$ and $\rho_k \to \rho_*$, we obtain that

$$T_p(x^*, 0) + \rho_*\|0\|^{p+1} \le T_p(x^*, s) + \rho_*\|s\|^{p+1}.$$

Since s satisfying $x^* + s \in D$ is arbitrary, we obtain the desired result. ∎

Recall that, if a point x^* is an unconstrained local minimizer of a function f, we have that such a point is p-stationary. A point $x \in \mathbb{R}^n$ is said to be q-order stationary (with $1 \le q \le p$) if it is $(q-1)$-order stationary and, for all $v \in \mathbb{R}^n$ such that $P_0(x, v) = \cdots = P_{j-1}(x, v) = 0$, one has that $P_j(x, v) \ge 0$. By convention, we say that every point $x \in \mathbb{R}^n$ is 0-order stationary and $P_0(x, v) = 0$. Note that p-stationarity may hold at points that are not minimizers at all. For example, $x = 0$ is p-stationary for the univariate function $-x^{p+1}$ for all $p \ge 1$ but is not a local minimizer for any p.

COROLLARY 3.1. *Assume that $q = 0$ (that is, $D = \mathbb{R}^n$ and the problem is unconstrained). Under the hypotheses of Theorem* 3.2*, the limit point x^* is m-order stationary for all $m \leq p$.*

Proof. By Theorem 3.2, $s = 0$ is q-order stationary for the function $T_p(x^*, s) + \rho_* \|s\|^{p+1}$. But the conditions of q-order stationarity of this function at $s = 0$ are the same as the ones of f at x^* since all their derivatives up to order p coincide. ∎

An m-order stationary condition for local minimization of $\varphi(x)$ subject to $x \in D$ is a property that involves derivatives up to order m of φ as well as properties of D and must be satisfied by any local minimizer of φ. Since, for all $m \leq p$, the m-order partial derivatives of f at x^* coincide with those of $T_p(x^*, s) + \rho_* \|s\|^{p+1}$ at $s = 0$, we may extend Corollary 3.1 to the constrained optimization case as follows.

COROLLARY 3.2. *Under the hypotheses of Theorem* 3.2*, the limit point x^* is m-order stationary for all $m \leq p$.*

4. Linearly constrained optimization. In this section, we consider the problem

$$\text{Minimize } f(x) \text{ subject to } x \in D, \tag{39}$$

where $D \subset \mathbb{R}^n$ is a polytope defined by

$$D = \{x \in \mathbb{R}^n \mid (a^i)^T x \leq b_i \text{ for all } i = 1, \ldots, q\}.$$

Algorithm 3.1 may be used as an independent algorithm to tackle the linearly constrained optimization problem (39) or it may be employed as the leaving-faces ingredient of an active-set strategy as in [3, 4, 12, 13, 14]. In [3, 4, 12, 13, 14], when the leaving-face criterion holds, the current face of a feasible set is abandoned using an iteration of the SPG method. However, the SPG method is also a competitive algorithm for solving large-scale problems with simple constraints. In the same sense, a single iteration of Algorithm 3.1 will be used here to leave faces, although Algorithm 3.1 may be considered as an independent algorithm for solving (39).

Given $I \subseteq \{1, \ldots, q\}$, we define the (open) face F_I by

$$F_I = \left\{x \in D \mid (a^i)^T x = b_i \text{ if } i \in I \text{ and } (a^i)^T x < b_i \text{ if } i \notin I\right\}.$$

Note that D is the union of the sets F_I for $I \subseteq \{1, \ldots, q\}$ and $I_1 \neq I_2$ implies that $F_{I_1} \cap F_{I_2} = \emptyset$. We define V_I as the smallest affine subspace in which a nonempty face F_I is contained and S_I as the corresponding parallel linear subspace; we define n_I as the dimension of V_I. Then, either a nonempty face F_I is a single point or V_I may be parameterized in terms of $n_I \geq 1$ "free" parameters $y \in \mathbb{R}^{n_I}$. Moreover, when a nonempty face F_I is not a single point, the interior of F_I is nonempty in terms of the variables y. Assume that the columns of $Q_I \in \mathbb{R}^{n \times n_I}$ are orthonormal and that S_I is parameterized as the set of linear combinations $Q_I y$ with $y \in \mathbb{R}^{n_I}$. Given $\hat{x} \in V_I$, every element $x \in V_I$ can be expressed in the form $x = \hat{x} + Q_I y$. Define

$$\hat{f}(\hat{x}; y) = f(\hat{x} + Q_I y).$$

Then, $\nabla \hat{f}(\hat{x}; y) = Q_I^T \nabla f(\hat{x} + Q_I y) = Q_I^T g(x)$ and $\nabla^2 \hat{f}(\hat{x}; y) = Q_I^T \nabla^2 f(\hat{x} + Q_I y) Q_I = Q_I^T H(x) Q_I$. In the algorithm described in the present section, if the current iterate x^k belongs to a face F_I and some criterion is satisfied, the computation of x^{k+1} consists in performing an iteration of Algorithm 2.1 for the minimization of $\hat{f}(x^k; y)$ within $\bar{F}_I$ (the closure of F_I), which is a polytope in the space $\mathbb{R}^{n_I}$.

We now consider the projection of $g(x)$ onto S_I, which is given by

$$g_I(x) = Q_I Q_I^T g(x),$$

and, for all $x \in F_I$, the projection of $g_I(x)$ onto $\bar{F}_I$, which is given by

$$\bar{g}_I(x) = P_{\bar{F}_I}(x - g_I(x)) - x.$$

Note that, if $x = \hat{x} + Q_I y$,

$$\|\bar{g}_I(x)\| \leq \|g_I(x)\| = \|\nabla \hat{f}(\hat{x}; y)\|,$$

where the inequality follows from the contraction property of projections and the equality holds by the definition of g_I. For any $x \in D$, since D being a polytope implies that $L(D, x) = D$, we define

$$g_P(x) = \nabla_D f(x) = P_D(x - g(x)) - x.$$

Given an iterate $x^k \in F_I$, the test that determines whether the current face F_I still deserves to be explored or should be abandoned involves a fraction $r \in (0, 1)$ and the quantities $\|\bar{g}_I(x^k)\|$ and $\|g_P(x^k)\|$. If

$$\|\bar{g}_I(x^k)\| \geq r\|g_P(x^k)\|$$

then, as already mentioned above, x^{k+1} is computed by performing an iteration of Algorithm 2.1 minimizing $\hat{f}(x^k; y)$ within $\bar{F}_I$. Otherwise, it is time to abandon the face and the new iterate x^{k+1} is computed by performing a single iteration of Algorithm 3.1 (with $p = 2$) applied to the minimization of $f(x)$ within D. The complete description of the algorithm follows.

ALGORITHM 4.1. *Let $x^0 \in D$, $\alpha > 0$, and $r \in (0, 1)$ be given. Set $k \leftarrow 0$.*

Step 1. *Let F_I be the face that contains x^k. Consider the test*

$$\|\bar{g}_I(x^k)\| \geq r\|g_P(x^k)\|. \tag{40}$$

Step 1.1. *If* (40) *holds, compute x^{k+1} performing one iteration of Algorithm* 2.1 *applied to the minimization of $\hat{f}(x^k; y)$ subject to $x^k + Q_I y \in \bar{F}_I$.*

Step 1.2. *If* (40) *does not hold, compute x^{k+1} performing one iteration of Algorithm* 3.1 *with $p = 2$ applied to the minimization of $f(x)$ subject to $x \in D$.*

Step 2. *Update $k \leftarrow k + 1$ and go to Step* 1.

Conditions similar to (40) for deciding to remain on faces have been used in [41] and works of Dostal [32, 33, 34, 35] and also by other authors (see [14] and the references therein). In the theorem below, there is some abuse of notation when the results of applying Algorithm 2.1 to the minimization of $f(x)$ subject to $x \in \Omega$ are considered valid for the application of Algorithm 2.1 to the minimization of $\hat{f}(x^k; y)$ subject to $x^k + Q_I y \in \bar{F}_I$. Of course, both problems are of the same type. Avoiding this abuse of notation would involve restating all the assumptions and results in section 2.

THEOREM 4.1. *Suppose that Assumptions* A2, A2 *(with $p = 2$),* A3, *and* A4 *hold and that the sequence $\{x^k\}$ is generated by Algorithm* 4.1. *Let $f_{\text{target}} \in \mathbb{R}$ and $\varepsilon_g = \varepsilon_d > 0$ be arbitrary. Then, the number of iterations k such that*

$$f(x^k) > f_{\text{target}} \quad \textit{and} \quad \|g_P(x^{k+1})\| > \varepsilon_g$$

is not greater than

(41)
$$\kappa = \left\lfloor (f(x_0) - f_{\text{target}})(q+2) \min\left\{ \frac{\alpha^{2/3}\varepsilon_g}{L+3\tau_2(L+\alpha)+\theta}, \frac{\alpha^{2/3} r \varepsilon_g}{L+3\max\{M,\tau_2(L+\alpha)\}}, \beta^{2/3}\varepsilon_g \right\}^{-3/2} \right\rfloor;$$

the number of functional evaluations per iterations is bounded above by

$$\max\left\{ \left\lfloor \log_{\tau_1}\left(\frac{\tau_2(L+\alpha)}{\rho_{\min}}\right)\right\rfloor + 1, J + \left\lfloor \log_{\tau_1}\left(\frac{\tau_2(L+\alpha)}{\hat{\rho}_0^{\text{ini}}}\right)\right\rfloor + 2 \right\} \tag{42}$$

when Algorithm 2.1 *is used with* $\nu = 0$*; and the total number of functional evaluations is bounded above by*

$$(\kappa+1)\left(\left\lfloor \log_{\tau_1}\left(\frac{\tau_2(L+\alpha)}{\rho_{\min}}\right)\right\rfloor + 1 + \log_{\tau_1}\left(\frac{\rho_{\max}}{\hat{\rho}_0^{\text{ini}}}\right) + \left|\log_{\tau_1}(\tau_0)\right| + (J+2)\right) \tag{43}$$

when Algorithm 2.1 *is used with* $\nu = 1$*. Finally, if* $\{f(x^k)\}$ *is bounded below, the number of iterations at which* (31) *holds is finite and*

$$\lim_{k\to\infty} \|g_P(x^{k+1})\| = 0. \tag{44}$$

Proof. If (40) does not hold, x^{k+1} is computed performing an iteration of Algorithm 3.1 with $p = 2$ and then, by Lemma 3.4, we have that

$$f(x^{k+1}) \le f(x^k) - \alpha\left(\frac{\|g_P(x^{k+1})\|}{L+3\tau_2\,(L+\alpha)+\theta}\right)^{3/2} \text{ or } f(x^{k+1}) \le f(x^k) - \beta\varepsilon^{3/2}. \tag{45}$$

Assume now that $x^k \in F_I$, that $x^{k+1} = x^k + Q_I\tilde{y}$ was computed by performing an iteration of Algorithm 2.1 applied to the minimization of $\hat{f}(x^k; y)$ subject to $x^k + Q_I y \in \bar{F}_I$, and that x^{k+1} belongs to F_I and not to $\bar{F}_I \setminus F_I$ (i.e., the boundary of F_I). Assume, in addition, that

$$\|\bar{g}_I(x^{k+1})\| \ge r\|g_P(x^{k+1})\|. \tag{46}$$

Then, by Lemma 2.5,

$$\hat{f}(x^k;\tilde{y}) \le \hat{f}(x^k;0) - \alpha\left(\frac{\|\nabla\hat{f}(x^k;\tilde{y})\|}{L+3\max\{M,\tau_2(L+\alpha)\}}\right)^{3/2}. \tag{47}$$

Since $\hat{f}(x^k;\tilde{y}) = f(x^{k+1})$, $\hat{f}(x^k;0) = f(x^k)$, $\|\nabla\hat{f}(x^k;\tilde{y})\| = \|g_I(x^{k+1})\| \ge \|\bar{g}_I(x^{k+1})\|$, and we are assuming that $\|\bar{g}_I(x^{k+1})\| \ge r\|g_P(x^{k+1})\|$, (47) implies that

$$f(x^{k+1}) \le f(x^k) - \alpha\left(\frac{r\|g_P(x^{k+1})\|}{L+3\max\{M,\tau_2(L+\alpha)\}}\right)^{3/2}. \tag{48}$$

Inequalities (45) and (48) show the decrease in the objective function that is obtained when, at iteration k, the new iterate x^{k+1} is computed, respectively, by (a) a single iteration of Algorithm 3.1 or (b) a single iteration of Algorithm 2.1 that computes an iterate that belongs to the interior of the current face and such that (46) holds. There are two cases that were not considered yet. Let F_I be the face to which x^k belongs. The first case corresponds to the case in which x^{k+1} is computed

by a single iteration of Algorithm 2.1 and x^{k+1} belongs to the boundary of the current face, i.e., $x^{k+1} \in \bar{F}_I \setminus F_I$. In this case, (47) may not hold and only a simple decrease of the form $f(x^{k+1}) < f(x^k)$ is granted. The second case corresponds to the case in which $x^{k+1} \in F_I$ is also computed by a single iteration of Algorithm 2.1 but

$$\|\bar{g}_I(x^{k+1})\| \not\geq r\|g_P(x^{k+1})\|. \tag{49}$$

In this case, (47) still holds, but due to (49), the functional decrease $O(\|g_P(x^{k+1})\|^{3/2})$ can not be established.

In order to cope with this state of facts, we will consider a sequence of $q+3$ consecutive iterates $x^\ell, x^{\ell+1}, \ldots, x^{\ell+q+2}$ aiming to establish that the decrease from $f(x^\ell)$ to $f(x^{\ell+q+2})$ is $O(\|g_P(x^{\ell+j})\|^{3/2})$ for some j between 1 and $q+2$. Since, by the definition of the algorithms, we have that $f(x^\ell) < f(x^{\ell+1}) < \cdots < f(x^{\ell+q+2})$, it would be enough to establish that there exists j, $1 \leq j \leq q+2$, such that the decrease from $f(x^{\ell+j-1})$ to $f(x^{\ell+j})$ is $O(\|g_P(x^{\ell+j})\|^{3/2})$. We will denote by $F_{I_{\ell+j}}$ the face to which $x^{\ell+j}$ belongs for $j = 0, 1, \ldots, q+2$. The analysis will be divided into three possible cases.

(a) There exists j, $1 \leq j \leq q+2$, such that $x^{\ell+j}$ was computed performing an iteration of Algorithm 3.1.

(b) There exists j, $1 \leq j \leq q+2$, such that $x^{\ell+j}$ was computed performing an iteration of Algorithm 2.1 and

$$x^{\ell+j} \in F_{I_{\ell+j-1}} \quad \text{and} \quad \|\bar{g}_{I_{\ell+j}}(x^{\ell+j})\| \geq r\|g_P(x^{\ell+j})\|. \tag{50}$$

(c) For all j, $1 \leq j \leq q+2$, $x^{\ell+j}$ was computed performing an iteration of Algorithm 2.1 and (50) does not hold, i.e.,

$$x^{\ell+j} \in \bar{F}_{I_{\ell+j-1}} \setminus F_{I_{\ell+j-1}} \quad \text{or} \quad \|\bar{g}_{I_{\ell+j}}(x^{\ell+j})\| \not\geq r\|g_P(x^{\ell+j})\|.$$

In cases (a) and (b), the desired decrease is given by (45) and (48), respectively. Let us analyze case (c). If $\|\bar{g}_{I_{\ell+j}}(x^{\ell+j})\| \not\geq r\|g_P(x^{\ell+j})\|$ for some $1 \leq j \leq q+1$ then, by the definition of Algorithm 4.1, the iterate $x^{\ell+j+1}$ is computed performing an iteration of Algorithm 3.1, which is a contradiction. Therefore, in case (c) we must have that for all j, $1 \leq j \leq q+1$, $x^{\ell+j}$ was computed performing an iteration of Algorithm 2.1 and $x^{\ell+j} \in \bar{F}_{I_{\ell+j-1}} \setminus F_{I_{\ell+j-1}}$. But, in this case, each iterate has at least one more active constraint than the previous iterate, meaning that $x^{\ell+q+2}$ should have at least $q+1$ active constraints, which is a contradiction because the problem being solved has q constraints. This means that case (c) can never occur and the desired decrease was established.

Up to now, we have proved that, for any given $q+3$ consecutive iterates x^ℓ, $x^{\ell+1}, \ldots, x^{\ell+q+2}$, it must follow that

$$f(x^{\ell+q+2}) \leq f(x^\ell) - \min\left\{ \frac{\alpha^{2/3}\|g_P(x^{\ell+j})\|}{L+3\tau_2(L+\alpha)+\theta}, \frac{\alpha^{2/3} r\|g_P(x^{\ell+j})\|}{L+3\max\{M,\tau_2(L+\alpha)\}}, \beta^{2/3}\varepsilon_g \right\}^{3/2} \tag{51}$$

for some j between 1 and $q+2$, from which (41) follows.

Let us now consider the number of functional evaluations. If Algorithm 2.1 is used with $\nu = 0$ then (42) follows from Theorems 2.1 and 3.1, which exhibit the bound on the number of functional evaluations per iteration of Algorithms 2.1 with $\nu = 0$ and 3.1, respectively. Assume now that Algorithm 2.1 is used with $\nu = 1$. Let $\kappa_1, \kappa_2 \geq 0$ satisfying $\kappa_1 + \kappa_2 \leq \kappa$ be the number of iterations of Algorithms 3.1 and 2.1

executed by Algorithm 4.1. Moreover, let $\kappa_{2,i} \geq 0$ be the number of consecutive iterations of Algorithm 2.1 that are executed in between the $(i-1)$th and the ith iterations of Algorithm 3.1 for $i = 2, \ldots, \kappa_1$, while $\kappa_{2,1} \geq 0$ and $\kappa_{2,\kappa_1+1} \geq 0$ are, respectively, the number of consecutive iterations of Algorithm 2.1 before the first and after the last iteration of Algorithm 3.1. Clearly, $\sum_{i=1}^{\kappa_1+1} \kappa_{2,i} = \kappa_2$. By Theorem 3.1, each iteration of Algorithm 3.1 consumes no more than

$$\left\lfloor \log_{\tau_1} \left(\frac{\tau_2(L+\alpha)}{\rho_{\min}} \right) \right\rfloor + 1 \tag{52}$$

functional evaluations, while, by Theorem 2.1, each group of consecutive $\kappa_{2,i}$ iterations of Algorithm 2.1 consumes no more than

$$\left\lfloor \log_{\tau_1} \left(\frac{\rho_{\max}}{\hat{\rho}_0^{\text{ini}}} \right) + \left| \log_{\tau_1}(\tau_0) \right| \kappa_{2,i} \right\rfloor + (J+2)(\kappa_{2,i}+1) \tag{53}$$

functional evaluations for $i = 1, \ldots, \kappa_1 + 1$. Summing (52) and (53), we have that the total number of functional evaluations consumed by Algorithm 4.1 is bounded above by

$$\kappa_1 \left(\left\lfloor \log_{\tau_1} \left(\frac{\tau_2(L+\alpha)}{\rho_{\min}} \right) \right\rfloor + 1 \right)$$

plus

$$\sum_{i=1}^{\kappa_1+1} \left(\left\lfloor \log_{\tau_1} \left(\frac{\rho_{\max}}{\hat{\rho}_0^{\text{ini}}} \right) + \left| \log_{\tau_1}(\tau_0) \right| \kappa_{2,i} \right\rfloor + (J+2)(\kappa_{2,i}+1) \right), \tag{54}$$

where $\kappa_2 = \sum_{i=1}^{\kappa_1+1} \kappa_{2,i}$ and $\kappa_1 + \kappa_2 \leq \kappa$. Distributing, it is easy to see that (54) is smaller than or equal to

$$(\kappa_1+1) \log_{\tau_1} \left(\frac{\rho_{\max}}{\hat{\rho}_0^{\text{ini}}} \right) + \left| \log_{\tau_1}(\tau_0) \right| \kappa_2 + (J+2)(\kappa_2 + \kappa_1 + 1) \tag{55}$$

and, therefore, (43) follows from (52), (55), and the fact that $\kappa_1 + \kappa_2 \leq \kappa$.

Finally, (44) follows from the boundedness of $\{f(x^k)\}$ and (51). ∎

5. Numerical experiments. We implemented Algorithms 2.1, 3.1 (for the $p = 2$ case only), and 4.1 in Fortran 90, for the particular case in which the feasible set D is given by $D = \{x \in \mathbb{R}^n \mid \ell \leq x \leq u\}$, where $\ell, u \in \mathbb{R}^n$, $\ell_i \leq u_i$ $(i = 1, \ldots, n)$, and ℓ_i and u_i may be $\mp\infty$ for some i, i.e., for box-constrained minimization. In Algorithm 2.1 (Step 2.1), a solution to (4) is computed using the method introduced in [15]. The increment s^{trial} in Algorithms 3.1 is computed by approximately solving (28) by the projected-gradient method (see, for example, [5, sect. 2.3]). It is worth noting that, in practice, we enforce the satisfaction of Assumption A4 with $s^k = s^{\text{trial}}$. With this objective, we impose a maximum number of iterations to the projected-gradient method applied to (28). If the projected-gradient method stops satisfying (32) with $s^k = s^{\text{trial}}$ and (26) holds then the step s^{trial} is accepted. If the projected-gradient method reaches the maximum number of iterations but (31) holds with $s^k = s^{\text{trial}}$ (or, equivalently, s^{trial} satisfies (27)) then the step s^{trial} is accepted. In any other case, we proceed, as already described in Algorithm 3.1, by increasing the regularizing parameter ρ. Note that, with these choices, Assumption A4 is satisfied.

In the numerical experiments, following [15], we considered $\alpha = 10^{-8}$, $M = 10^3$, $\tau_1 = 2$, $\tau_2 = 50$, and $\rho_{\min} = 10^{-6}$ in Algorithm 2.1. Values of $J \in \{0, 1, 10\}$,

which correspond to no magical steps, a single magical step per iteration, and at most 10 magical steps per iterations, respectively, will be tested. In Algorithm 3.1, we arbitrarily considered $\alpha = 10^{-8}$, $\rho_{\min} = 10^{-6}$, $\tau_1 = \tau_2 = 10$, $\theta = 1$, and $\beta = 1$. In Algorithm 4.1, we arbitrarily considered $\alpha = 10^{-8}$ and $r = 0.1$. As a stopping criterion for Algorithms 3.1 and 4.1, we considered the condition

$$\|g_P(x^k)\|_\infty \leq \varepsilon \tag{56}$$

with $\varepsilon = 10^{-6}$. It should be noted that none of these parameters were subject to tuning at all. All of them were chosen because they seemed to be "natural choices" and the intention of the numerical experiments below is not to deliver the most robust or efficient version of the proposed method but to illustrate its practical behavior in terms of consumption of functional evaluations.

We performed numerical experiments considering all the 105 bound-constrained problems from the CUTEst [42] collection (version 1.1, June 17, 2013) with less than 10,000 variables (considering the default dimension of the problems). All tests were conducted on a computer with 3.5 GHz Intel Core i7 processor and 16GB 1600 MHz DDR3 RAM memory, running OS X Yosemite (version 10.10.5). Codes were compiled by the GFortran Fortran compiler of GCC (version 7.2.0) with the -O3 optimization directive enabled.

The focus of the numerical experiments is to evaluate the performance of the proposed methods in terms of number of functional evaluations. With this purpose, we evaluated Algorithms 3.1 and 4.1 with $J \in \{0, 1, 10\}$. Preliminary numerical experiments showed that Algorithm 4.1 is more efficient when Algorithm 2.1 uses the strategy given by $\nu = 1$ for updating the regularization parameter. (A comparison of these two strategies in the context of unconstrained minimization can be found in [8].) It should be noted that both strategies ($\nu = 0$ and $\nu = 1$) consider as first trial at every iteration the null regularization parameter. For this reason, the efficiency of the strategy given by $\nu = 1$ is not related to the lack of a lower bound on the regularization parameter, but it is due to the fact that each iteration starts with a regularization parameter that is close to the one that was successful in the previous iteration, saving functional evaluations. Algorithm 4.1 was also compared with the active-set method for bound-constrained minimization introduced in [3] named BETRA. As already mentioned in the introduction, BETRA uses a trust-region strategy within the faces and spectral projected gradients for leaving faces. Therefore, Algorithm 4.1 could be seen as the method as close as possible to BETRA that possesses worst-case evaluation complexity results and, thus, the results of this comparison could be interpreted as an answer to the question as to whether it is profitable or not to develop practical methods possessing worst-case evaluation complexity theoretical results.

In order to make the experiments affordable, a CPU time limit of one hour was applied to each pair algorithm/problem. Since the analysis of the performance was based on functional evaluations, problems in which at least one of the methods fails in satisfying the stopping criterion (56) within the CPU time limit will be (reported and) eliminated from the comparison.

We first analyze the performance of Algorithm 4.1 varying $J \in \{0, 1, 10\}$. Algorithm 4.1 with $J \in \{0, 1, 10\}$ satisfied the stopping criterion (56) within the CPU time limit in 90, 97, and 97 problems, respectively. Eliminating the problems in which at least one of the variants failed (in satisfying the stopping criterion within the CPU time limit), we obtain a subset with 90 problems. (Detailed information regarding the performance of each method on each problem can be found at http://www.ime.usp.br/~egbirgin/.) For a given problem, let f_1, f_2, and f_3 be the

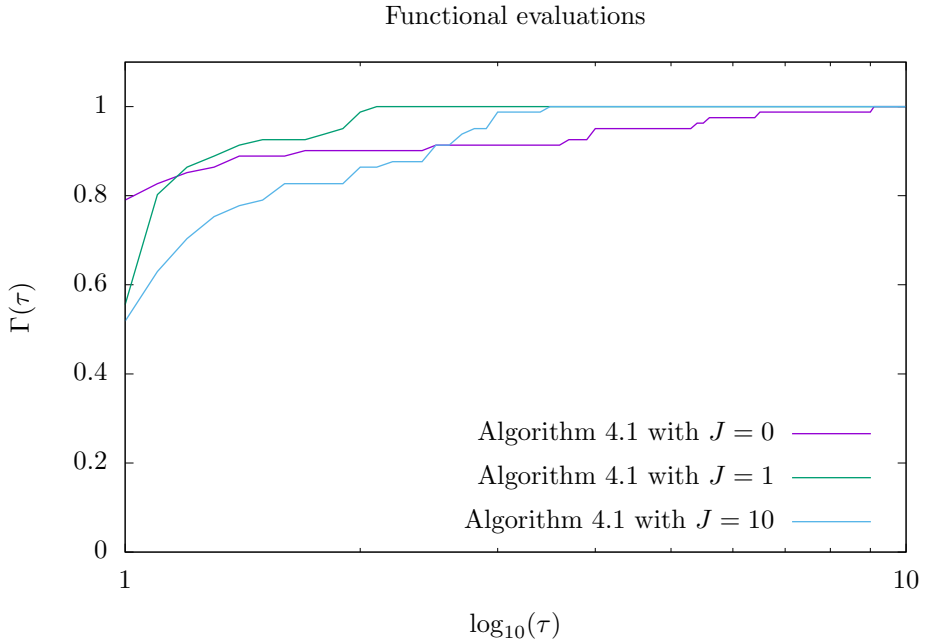

FIG. 1. *Performance profile analyzing the influence of the magical steps (varying* $J \in \{0, 1, 10\}$*) in the efficiency of Algorithm* 4.1.

value of the objective function at the final iterate delivered by each variant of Algorithm 4.1, respectively. Following [11], we will say that the methods being compared found *equivalent* solutions if

$$\frac{f_i - f_{\text{best}}}{\max\{1, |f_{\text{best}}|\}} \leq 10^{-4} \text{ for } i = 1, 2, 3,$$

where $f_{\text{best}} = \min\{f_1, f_2, f_3\}$. Applying this criterion to the 90 problems in which the three variants of Algorithm 4.1 satisfied the stopping criterion within the imposed CPU time limit, we obtain that they found equivalent solutions in 81 problems. The efficiencies of the variants are compared using these 81 problems in the performance profile displayed in Figure 1 (see color figures in the online version), while Table 1 shows the details of the performance of the methods in the other $105 - 81 = 24$ problems (in which at least one of the methods did not satisfy the stopping criterion within the imposed CPU time limit or the three methods satisfied the stopping criterion but they found nonequivalent solutions). In the table, "SC" stands for stopping criterion, "CO" means that the stopping criterion (56) was satisfied, "TE" means that the CPU time limit was achieved, and "UN" means that the method stopped because an iterate x^k satisfying $f(x^k) \leq -10^{10}$ was found (suggesting that the objective function is unbounded from below within the feasible region D). It is not easy to make conclusions on the robustness of the methods from the figures in the table that correspond to problems in which at least one of the methods did not satisfy the stopping criterion within the limit imposed on the CPU time. This is because, doing that, we take the risk of attributing lack of robustness to a method due to something that, in fact, may be lack of efficiency. Therefore, we restrict ourself to

TABLE 1

Performance of Algorithm 4.1 *with* $J \in \{0, 1, 10\}$ *in the problems in which at least one of the three variants did not satisfy the stopping criterion within the CPU time limit or they found nonequivalent solutions.*

Problem	Algorithm 4.1 with $J = 0$			Algorithm 4.1 with $J = 1$			Algorithm 4.1 with $J = 10$		
	$f(x^k)$	$\|g_P(x^k)\|_\infty$	SC	$f(x^k)$	$\|g_P(x^k)\|_\infty$	SC	$f(x^k)$	$\|g_P(x^k)\|_\infty$	SC
BIGGSB1	1.543684e−02	8.9e−07	CO	1.500000e−02	2.2e−16	CO	1.500000e−02	2.2e−16	CO
BQPGAUSS	-3.529664e−01	3.0e−02	TE	-3.595735e−01	1.1e−02	TE	-3.595015e−01	9.1e−02	TE
CHENHARK	-1.690100e+00	3.0e−01	TE	-1.999996e+00	3.5e−04	TE	-1.999995e+00	3.8e−04	TE
HADAMALS	1.136748e+02	4.3e−12	CO	1.259669e+02	3.0e−09	CO	1.805902e+02	1.5e−07	CO
HARKERP2	1.992050e+09	3.2e+01	TE	-5.000000e−01	4.3e−13	CO	-5.000000e−01	4.3e−13	CO
MAXLIKA	1.136307e+03	8.4e−08	CO	1.136307e+03	1.2e−07	CO	1.149346e+03	1.6e−10	CO
PALMER4	2.424016e+03	7.1e−07	CO	2.285383e+03	2.6e−12	CO	2.285383e+03	2.6e−12	CO
PALMER5A	2.594999e−02	2.8e+00	TE	2.594999e−02	2.8e+00	TE	2.594999e−02	2.8e+00	TE
PALMER5E	2.081326e−02	4.9e+00	TE	2.081326e−02	4.9e+00	TE	2.081326e−02	4.9e+00	TE
PALMER8E	6.339306e−03	3.3e−08	CO	6.339306e−03	3.3e−08	CO	6.340519e−01	1.4e−10	CO
POWELLBC	3.102475e+05	4.5e−08	CO	3.102862e+05	6.9e−11	CO	3.348701e+05	1.4e−07	CO
QRTQUAD	-2.648253e+11	1.0e+01	UN	-2.648253e+11	1.0e+01	UN	-2.648253e+11	1.0e+01	UN
S368	-7.500000e−01	5.4e−09	CO	-1.000000e+00	7.8e−12	CO	-1.000000e+00	1.8e−12	CO
SCOND1LS	6.862749e+04	7.1e+02	TE	6.862749e+04	7.1e+02	TE	6.862749e+04	7.1e+02	TE
SINEALI	-9.978692e+04	1.6e−12	CO	-9.960170e+04	4.6e−09	CO	-9.960170e+04	4.6e−09	CO
TORSION2	-1.899696e−01	9.0e−04	TE	-4.302758e−01	2.2e−16	CO	-4.302758e−01	2.2e−16	CO
TORSION4	-3.032020e−01	1.9e−03	TE	-1.216956e+00	1.7e−16	CO	-1.216956e+00	1.7e−16	CO
TORSION6	-4.674420e−01	3.8e−03	TE	-2.863378e+00	1.7e−16	CO	-2.863378e+00	1.7e−16	CO
TORSIONB	-3.429606e−01	6.3e−04	TE	-4.182962e−01	3.4e−09	CO	-4.182962e−01	3.4e−09	CO
TORSIOND	-6.029582e−01	1.8e−03	TE	-1.204209e+00	2.2e−16	CO	-1.204209e+00	2.2e−16	CO
TORSIONF	-9.292051e−01	3.8e−03	TE	-2.850248e+00	2.2e−16	CO	-2.850248e+00	2.2e−16	CO
WALL10	-3.198632e+05	9.9e−02	TE	-3.171402e+05	9.7e−02	TE	-4.559541e+05	2.5e−05	TE
WALL20	-1.339585e+01	8.4e−01	TE	-1.339585e+01	8.4e−01	TE	-1.339585e+01	8.4e−01	TE
WEEDS	2.587277e+00	4.1e−11	CO	2.587277e+00	4.1e−11	CO	9.205435e+03	8.5e−14	CO

analyzing problems BIGGSB1, HADAMALS, MAXLIKA, PALMER4, PALMER8E, POWELLBC, S368, SINEALI, and WEEDS, which are the nine problems in which the three methods satisfied the stopping criterion but found different solutions. In those problems, Algorithm 4.1 with $J \in \{0, 1, 10\}$ found a final iterate with a smaller objective functional value (than the one found by the other variants) in five, six, and four problems, respectively. The conclusion is that Algorithm 4.1 with $J = 1$ appears to be the most robust and efficient version of Algorithm 4.1 and that it performs only a few unsuccessful magical steps that require extra functional evaluations.

We now compare the performances of Algorithms 3.1 and 4.1 with $J = 1$. Algorithm 3.1 satisfied the stopping criterion (56) within the CPU time limit in 87 problems (recall that this number is 97 for Algorithm 4.1 with $J = 1$). Both algorithms succeeded in satisfying the stopping criterion within the CPU time limit in 87 problems and, among them, they found equivalent solutions in 79 problems. The efficiency of both algorithms is compared considering these 79 problems in the performance profile displayed in Figure 2 (see color figures in the online version). Details of the performance of the methods in the other $105 - 79 = 26$ problems are given in Table 2. In the table, it can be seen that both methods satisfied the stopping criterion but found nonequivalent solutions in the following eight problems: CAMEL6, CHEBYQAD, EG1, HADAMALS, PALMER3, PALMER3E, PALMER4, and PALMER7A. Algorithm 4.1 found smaller functional values in six cases and larger functional values in two problems. The conclusion is that Algorithm 4.1 (with $J = 1$) appears to be more efficient and robust than Algorithm 3.1.

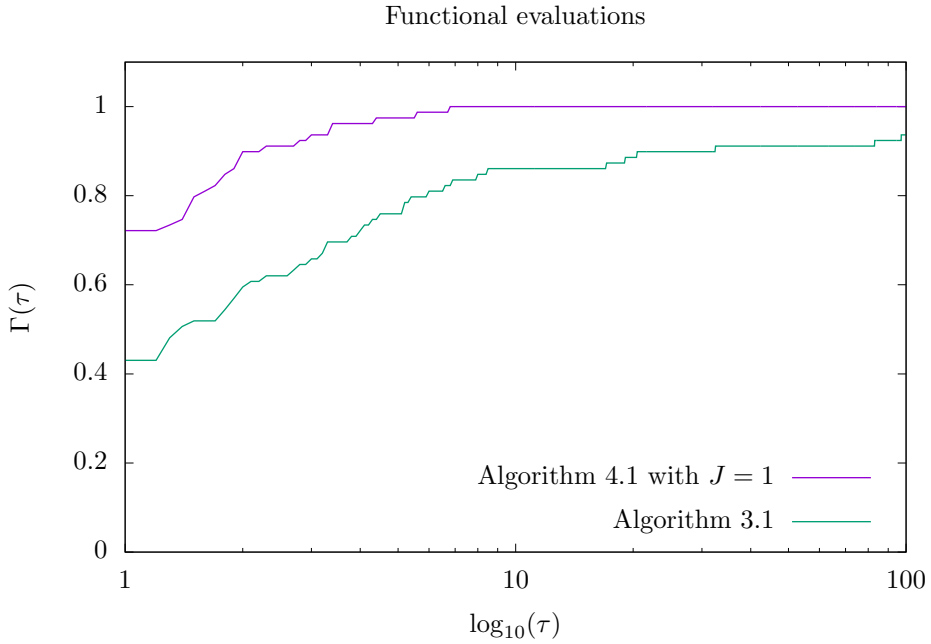

FIG. 2. *Performance profile comparing the efficiency of Algorithms* 3.1 *and* 4.1 *with* $J = 1$.

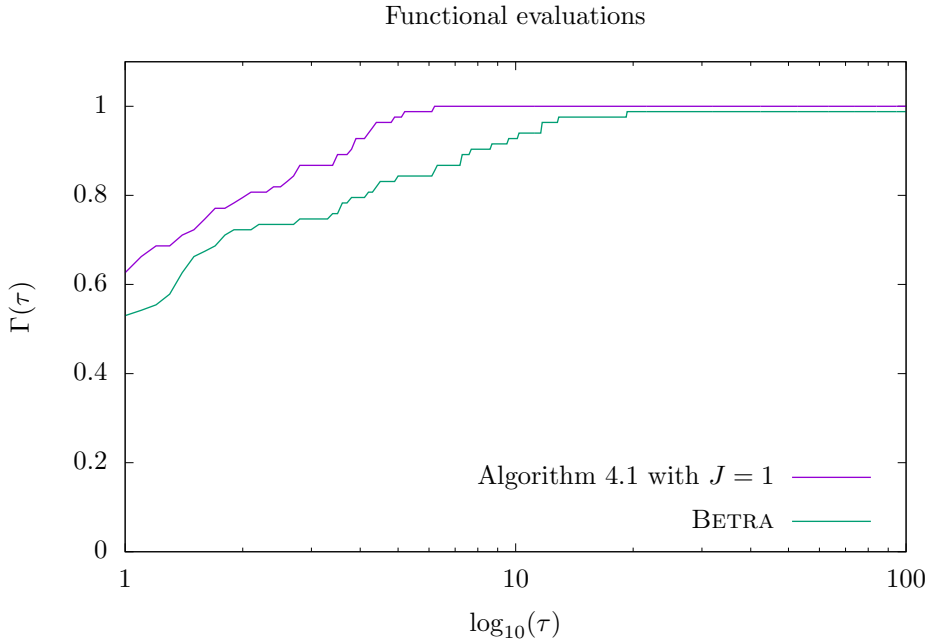

FIG. 3. *Performance profile comparing the efficiency of Algorithm* 4.1 *with* $J = 1$ *and* BETRA.

TABLE 2

Performance of Algorithms 3.1 *and* 4.1 *with* $J = 1$ *in the problems in which at least one of them did not satisfy the stopping criterion within the CPU time limit or they found nonequivalent solutions.*

Problem	Algorithm 3.1			Algorithm 4.1 with $J = 1$		
	$f(x^k)$	$\|g_P(x^k)\|_\infty$	SC	$f(x^k)$	$\|g_P(x^k)\|_\infty$	SC
3PK	1.724077e+00	5.3e−04	TE	1.720119e+00	1.6e−09	CO
BIGGSB1	1.924139e−02	2.9e−05	TE	1.500000e−02	2.2e−16	CO
BQPGAUSS	-2.798142e−01	1.0e−01	TE	-3.595735e−01	1.1e−02	TE
CAMEL6	-2.154638e−01	4.3e−11	CO	-1.031628e+00	2.1e−07	CO
CHEBYQAD	1.030983e−02	9.9e−07	CO	4.513555e−03	5.4e−07	CO
CHENHARK	-1.999541e+00	8.0e−05	TE	-1.999996e+00	3.5e−04	TE
EG1	-1.429307e+00	8.6e−13	CO	-1.132801e+00	6.3e−08	CO
GRIDGENA	2.352000e+04	8.4e−05	TE	2.352000e+04	4.8e−11	CO
HADAMALS	1.526408e+02	6.2e−09	CO	1.259669e+02	3.0e−09	CO
PALMER1E	8.352742e−04	2.8e−06	TE	8.352322e−04	4.4e−08	CO
PALMER2E	2.917189e−03	2.2e−03	TE	6.113360e−02	1.2e−09	CO
PALMER3	2.416980e+03	2.6e−08	CO	2.265958e+03	6.8e−07	CO
PALMER3E	5.074106e−05	6.8e−07	CO	7.086597e−02	3.3e−07	CO
PALMER4	2.424016e+03	1.1e−08	CO	2.285383e+03	2.6e−12	CO
PALMER5A	1.452576e−01	8.2e−04	TE	2.594999e−02	2.8e+00	TE
PALMER5B	2.116375e−02	5.1e−04	TE	9.752418e−03	1.5e−07	CO
PALMER5E	3.710294e−02	3.2e−05	TE	2.081326e−02	4.9e+00	TE
PALMER7A	2.792939e+01	5.8e−07	CO	1.033486e+01	2.8e−08	CO
PALMER7E	1.015391e+01	7.9e−06	TE	1.015390e+01	7.0e−09	CO
POWELLBC	Infinity	1.0e+00	TE	3.102862e+05	6.9e−11	CO
QR3DLS	1.687343e−02	2.5e−03	TE	1.749783e−20	2.7e−09	CO
QRTQUAD	-9.460519e+09	4.9e+04	TE	-2.648253e+11	1.0e+01	UN
SCOND1LS	4.231559e+05	7.1e+02	TE	6.862749e+04	7.1e+02	TE
SINEALI	-9.987336e+04	3.5e−06	TE	-9.960170e+04	4.6e−09	CO
WALL10	-4.992176e+00	1.8e−01	TE	-3.171402e+05	9.7e−02	TE
WALL20	-1.357148e+01	6.7e−01	TE	-1.339585e+01	8.4e−01	TE

Finally, we compare the performances of Algorithm 4.1 with $J = 1$ and BETRA. BETRA satisfied the stopping criterion (56) within the CPU time limit in 96 problems (recall that this number is 97 for Algorithm 4.1 with $J = 1$). Both algorithms succeeded in satisfying the stopping criterion within the CPU time limit in 92 problems and, among them, they found equivalent solutions in 83 problems. The efficiency of both algorithms is compared considering these 83 problems in the performance profile displayed in Figure 3 (see color figures in the online version). Details of the performance of the methods in the other $105 - 83 = 22$ problems are given in Table 3. In this table, "LP" in the column related to the stopping criterion of BETRA means "lack of progress." In the table, it can be seen that both methods satisfied the stopping criterion but found nonequivalent solutions in the following nine problems: CHEBYQAD, DECONVB, HADAMALS, MAXLIKA, PALMER2E, PALMER3E, PALMER7A, S368, and SINEALI. Algorithm 4.1 found smaller functional values in six cases and larger functional values in three problems. The conclusion is that Algorithm 4.1 (with $J = 1$) appears to be more efficient and robust than BETRA.

TABLE 3

Performance of Algorithm 4.1 with $J = 1$ and BETRA in the problems in which at least one of them did not satisfy the stopping criterion within the CPU time limit or they found nonequivalent solutions.

Problem	Algorithm 4.1 with $J = 1$			BETRA		
	$f(x^k)$	$\|g_P(x^k)\|_\infty$	SC	$f(x^k)$	$\|g_P(x^k)\|_\infty$	SC
BQPGAUSS	-3.595735e−01	1.1e−02	TE	-3.625778e−01	7.2e−13	CO
CHEBYQAD	4.513555e−03	5.4e−07	CO	9.055436e−03	5.9e−07	CO
CHENHARK	-1.999996e+00	3.5e−04	TE	-2.000000e+00	1.2e−12	CO
DECONVB	7.062747e−09	3.6e−07	CO	8.638251e−03	6.5e−09	CO
EXPLIN2	-7.199883e+07	9.8e−12	CO	-7.199883e+07	1.9e−04	LP
HADAMALS	1.259669e+02	3.0e−09	CO	1.596622e+02	6.4e−11	CO
LINVERSE	6.810000e+02	2.8e−09	CO	6.820000e+02	2.3e−06	LP
MAXLIKA	1.136307e+03	1.2e−07	CO	1.149346e+03	7.6e−07	CO
PALMER2E	6.113360e−02	1.2e−09	CO	2.065035e−04	4.1e−07	CO
PALMER3E	7.086597e−02	3.3e−07	CO	5.074105e−05	9.1e−08	CO
PALMER4	2.285383e+03	2.6e−12	CO	2.327886e+03	1.5e+00	LP
PALMER5A	2.594999e−02	2.8e+00	TE	2.137729e−02	9.2e−07	CO
PALMER5E	2.081326e−02	4.9e+00	TE	2.071594e−02	2.9e−08	CO
PALMER7A	1.033486e+01	2.8e−08	CO	2.792939e+01	5.8e−11	CO
PALMER7E	1.015390e+01	7.0e−09	CO	1.015390e+01	1.6e−03	LP
POWELLBC	3.102862e+05	6.9e−11	CO	6.740747e+05	1.0e+00	TE
QRTQUAD	-2.648253e+11	1.0e+01	UN	-2.648567e+11	1.0e+01	LP
S368	-1.000000e+00	7.8e−12	CO	-7.500000e−01	4.9e−09	CO
SCOND1LS	6.862749e+04	7.1e+02	TE	1.001405e+02	3.2e+01	TE
SINEALI	-9.960170e+04	4.6e−09	CO	-9.989947e+04	7.6e−09	CO
WALL10	-3.171402e+05	9.7e−02	TE	-4.559540e+05	4.8e−01	LP
WALL20	-1.339585e+01	8.4e−01	TE	-4.322414e+06	2.0e+01	LP

6. Conclusions. In this paper we introduced the following algorithms: (1) Algorithm 2.1 addresses the minimization of f with general constraints finding an interior point with sufficiently small gradient or a point on the boundary at which the function decreases; (2) Algorithm 3.1 aims to minimize a function on an arbitrary domain using a high-order Taylor-like model at each iteration; and (3) Algorithm 4.1 minimizes a function with linear constraints employing Algorithm 2.1 within the faces and a single iteration of Algorithm 3.1 (with $p = 2$) for discarding active constraints.

Algorithm 2.1 achieves the goal of finding an interior point with gradient norm smaller than ε or a sufficiently good point on the boundary with complexity $O(\varepsilon^{-3/2})$. Algorithm 3.1 finds a point whose "continuous projected gradient" norm is smaller than ε with complexity $O(\varepsilon^{-(p+1)/p})$. Algorithm 4.1 finds a continuous projected gradient norm smaller than ε, also with complexity $O(\varepsilon^{-3/2})$.

The comparison of Algorithm 3.1 (with $p = 2$) against Algorithm 4.1 for solving box-constrained optimization problems reveals that Algorithm 4.1 is more efficient and robust, while the comparison of Algorithm 4.1 against BETRA shows that Algorithm 4.1 is also more efficient and robust than BETRA, suggesting that developing practical methods that possess worst-case evaluation complexity results may be profitable.

Acknowledgments. We are indebted to Philippe Toint, Geovani Grapiglia, and the referees for insightful comments, suggestions, and discussions regarding the final version of this paper.

REFERENCES

[1] R. Andreani, E. G. Birgin, J. M. Martínez, and J. Y. Yuan, *Spectral projected gradient and variable metric methods for optimization with linear inequalities*, IMA J. Numer. Anal., 25 (2005), pp. 221–252, doi:10.1093/imanum/drh020.

[2] R. Andreani, G. Haeser, and J. M. Martínez, *On sequential optimality conditions for smooth constrained optimization*, Optimization, 60 (2011), pp. 627–641, doi:10.1080/02331930903578700.

[3] M. Andretta, E. G. Birgin, and J. M. Martínez, *Practical active-set euclidian trust-region method with spectral projected gradients for bound-constrained minimization*, Optimization, 54 (2005), pp. 305–325, doi:10.1080/02331930500100270.

[4] M. Andretta, E. G. Birgin, and J. M. Martínez, *Partial spectral projected gradient method with active-set strategy for linearly constrained optimization*, Numer. Algorithms, 53 (2010), pp. 23–52, doi:10.1007/s11075-009-9289-9.

[5] D. P. Bertsekas, *Nonlinear Programming*, 2nd ed., Athena Scientific, Belmont, MA, 1999.

[6] T. Bianconcini, G. Liuzzi, B. Morini, and M. Sciandrone, *On the use of iterative methods in cubic regularization for unconstrained optimization*, Comput. Optim. Appl., 60 (2015), pp. 35–57, doi:10.1007/s10589-014-9672-x.

[7] T. Bianconcini and M. Sciandrone, *A cubic regularization algorithm for unconstrained optimization using line search and nonmonotone techniques*, Optim. Methods Softw., 31 (2016), pp. 1008–1035, doi:10.1080/10556788.2016.1155213.

[8] E. G. Birgin, J. L. Gardenghi, J. M. Martínez, and S. A. Santos, *Is it Worth Using Third-Order Models with Fourth-Order Regularization for Unconstrained Optimization?*, Technical report MCDO131017, Institute of Mathematics, Statistics, and Scientific Computing, University of Campinas, Brazil, 2017.

[9] E. G. Birgin, J. L. Gardenghi, J. M. Martínez, S. A. Santos, and P. L. Toint, *Evaluation complexity for nonlinear constrained optimization using unscaled KKT conditions and high-order models*, SIAM J. Optim., 26 (2016), pp. 951–967, doi:10.1137/15M1031631.

[10] E. G. Birgin, J. L. Gardenghi, J. M. Martínez, S. A. Santos, and P. L. Toint, *Worst-case evaluation complexity for unconstrained nonlinear optimization using high-order regularized models*, Math. Program., 163 (2017), pp. 359–368, doi:10.1007/s10107-016-1065-8.

[11] E. G. Birgin and J. M. Gentil, *Evaluating bound-constrained minimization software*, Comput. Optim. Appl., 53 (2012), pp. 347–373, doi:10.1007/s10589-012-9466-y.

[12] E. G. Birgin and J. M. Martínez, *A box-constrained optimization algorithm with negative curvature directions and spectral projected gradients*, in Topics in Numerical Analysis, Computing Suppl. 15, G. Alefeld and X. Chen, eds., Springer, Vienna, 2001, pp. 49–60, doi:10.1007/978-3-7091-6217-0_5.

[13] E. G. Birgin and J. M. Martínez, *Large-scale active-set box-constrained optimization method with spectral projected gradients*, Comput. Optim. Appl., 23 (2002), pp. 101–125, doi:10.1023/A:1019928808826.

[14] E. G. Birgin and J. M. Martínez, *Practical Augmented Lagrangian Methods for Constrained Optimization*, Fundam. Algorithms 10, SIAM, Philadelphia, PA, 2014, doi:10.1137/1.9781611973365.

[15] E. G. Birgin and J. M. Martínez, *The use of quadratic regularization with a cubic descent condition for unconstrained optimization*, SIAM J. Optim., 27 (2017), pp. 1049–1074, doi:10.1137/16M110280X.

[16] E. G. Birgin, J. M. Martínez, and M. Raydan, *Nonmonotone spectral projected gradient methods on convex sets*, SIAM J. Optim., 10 (2000), pp. 1196–1211, doi:10.1137/S1052623497330963.

[17] E. G. Birgin, J. M. Martínez, and M. Raydan, *Algorithm 813: SPG – software for convex-constrained optimization*, ACM Trans. Math. Software, 27 (2001), pp. 340–349, doi:10.1145/502800.502803.

[18] E. G. Birgin, J. M. Martínez, and M. Raydan, *Inexact spectral projected gradient methods on convex sets*, IMA J. Numer. Anal., 23 (2003), pp. 539–559, doi:10.1093/imanum/23.4.539.

[19] E. G. Birgin, J. M. Martínez, and M. Raydan, *Spectral projected gradient methods: Review and perspectives*, J. Stat. Softw., 60 (2014), doi:10.18637/jss.v060.i03.

[20] C. Cartis, N. I. M. Gould, and P. L. Toint, *On the complexity of steepest descent, Newton's*

and regularized Newton's methods for nonconvex unconstrained optimization, SIAM J. Optim., 20 (2010), pp. 2833–2852, doi:10.1137/090774100.

[21] C. Cartis, N. I. M. Gould, and P. L. Toint, *Adaptive cubic regularization methods for unconstrained optimization. Part* I*: Motivation, convergence and numerical results*, Math. Program., 127 (2011), pp. 245–295, doi:10.1007/s10107-009-0286-5.

[22] C. Cartis, N. I. M. Gould, and P. L. Toint, *Adaptive cubic regularization methods for unconstrained optimization. Part* II*: Worst-case function and derivative complexity*, Math. Program., 130 (2011), pp. 295–319, doi:10.1007/s10107-009-0337-y.

[23] C. Cartis, N. I. M. Gould, and P. L. Toint, *An adaptive cubic regularization algorithm for nonconvex optimization with convex constraints and its function-evaluation complexity*, IMA J. Numer. Anal., 32 (2012), pp. 1662–1695, doi:10.1093/imanum/drr035.

[24] C. Cartis, N. I. M. Gould, and P. L. Toint, *Complexity bounds for second-order optimality in unconstrained optimization*, J. Complexity, 28 (2012), pp. 93–108, doi:10.1016/j.jco.2011.06.001.

[25] C. Cartis, N. I. M. Gould, and P. L. Toint, *Optimal Newton-Type Methods for Nonconvex Optimization*, Technical report naXys-17-2011, Namur Center for Complex Systems (naXys), University of Namur, Belgium, 2012.

[26] C. Cartis, N. I. M. Gould, and P. L. Toint, *Evaluation Complexity Bounds for Smooth Constrained Nonlinear Optimization using Scaled KKT Conditions and High-Order Models*, Technical report naXys-11-2015(R1), Namur Center for Complex Systems (naXys), University of Namur, Belgium, 2015.

[27] C. Cartis, N. I. M. Gould, and P. L. Toint, *Universal regularization methods – varying the power, the smoothness and the accuracy*, Optim. Methods Softw., to appear.

[28] A. R. Conn, N. I. M. Gould, and P. L. Toint, *Global convergence of a class of trust region algorithms for optimization with simple bounds*, SIAM J. Numer. Anal., 25 (1988), pp. 433–460, doi:10.1137/0725029.

[29] A. R. Conn, N. I. M. Gould, and P. L. Toint, *Trust Region Methods*, Society for Industral and Applied Mathematics, Philadelphia, PA, 2000, doi:10.1137/1.9780898719857.

[30] F. E. Curtis, D. P. Robinson, and M. Samadi, *An Inexact Regularized Newton Framework with a Worst-Case Iteration Complexity of* $o(\epsilon^{-3/2})$ *for Nonconvex Optimization*, preprint, arXiv:1708.00475, 2017.

[31] F. E. Curtis, D. P. Robinson, and M. Samadi, *A trust-region algorithm with a worst-case iteration complexity of* $O(\varepsilon^{-3/2})$, Math. Program., 162 (2017), pp. 1–32, doi:10.1007/s10107-016-1026-2.

[32] M. Domorádová and Z. Dostál, *Projector preconditioning for partially bound-constrained quadratic optimization*, Numer. Linear Algebra Appl., 14 (2007), pp. 791–806, doi:10.1002/nla.555.

[33] Z. Dostál, *Directions of Large Decrease and Quadratic Programming*, Technical report, Department of Applied Mathematics, Technical University of Ostrava, Czech Republic, 1994.

[34] Z. Dostál, *Box constrained quadratic programming with proportioning and projections*, SIAM J. Optim., 7 (1997), pp. 871–887, doi:10.1137/S1052623494266250.

[35] Z. Dostál, *Optimal Quadratic Programming Algorithms*, Springer Optim. Appl. 23, Springer, Heidelberg, Berlin, New York, 2009, doi:10.1007/b138610.

[36] J. C. Dunn, *Newtons method and the Goldstein step-length rule for constrained minimization problems*, SIAM J. Control Optim., 18 (1980), pp. 659–674, doi:10.1137/0318050.

[37] J. C. Dunn, *Global and asymptotic convergence rate estimates for a class of projected gradient processes*, SIAM J. Control Optim., 19 (1981), pp. 368–400, doi:10.1137/0319022.

[38] J. C. Dunn, *On the convergence of projected gradient processes to singular critical points*, J. Optim. Theory Appl., 55 (1987), pp. 203–216, doi:10.1007/BF00939081.

[39] J. P. Dussault, *Arc$_q$: A new adaptive regularization by cubics*, Optim. Methods Softw., 33 (2017), pp. 322–335, doi:10.1080/10556788.2017.1322080.

[40] R. Fletcher, *Practical Methods of Optimization*, 2nd ed., John Wiley and Sons, London, 1987, doi:10.1002/9781118723203.

[41] A. Friedlander and J. M. Martínez, *On the numerical solution of bound constrained optimization problems*, RAIRO Oper. Res., 23 (1989), pp. 319–341.

[42] N. I. M. Gould, D. Orban, and P. L. Toint, *CUTEst: A constrained and unconstrained testing environment with safe threads for mathematical optimization*, Comput. Optim. Appl., 60 (2014), pp. 545–557, doi:10.1007/s10589-014-9687-3.

[43] G. N. Grapiglia and Y. Nesterov, *Regularized Newton methods for minimizing functions with Hölder continuous hessians*, SIAM J. Optim., 27 (2017), pp. 478–506, doi:10.1137/16M1087801.

[44] G. N. Grapiglia, J.-Y. Yuan, and Y.-X. Yuan, *On the convergence and worst-case com-*

plexity of trust-region and regularization methods for unconstrained optimization, Math. Program., 152 (2015), pp. 491–520, doi:10.1007/s10107-014-0794-9.

[45] A. Griewank, *The Modification of Newton's Method for Unconstrained Optimization by Bounding Cubic Terms*, Technical report NA/12, Department of Applied Mathematics and Theoretical Physics, University of Cambridge, England, 1981.

[46] J. M. Martínez, *On high-order model regularization for constrained optimization*, SIAM J. Optim., 27 (2017), pp. 24472458, doi.org/10.1137/17M1115472.

[47] J. M. Martínez and M. Raydan, *Cubic-regularization counterpart of a variable-norm trust-region method for unconstrained minimization*, J. Global Optim., 68 (2017), pp. 367–385, doi:10.1007/s10898-016-0475-8.

[48] J. M. Martínez and B. F. Svaiter, *A practical optimality condition without constraint qualifications for nonlinear programming*, J. Optim. Theory Appl., 118 (2003), pp. 117–133, doi:10.1023/A:1024791525441.

[49] J. L. Morales and J. Nocedal, *Remark on Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound constrained optimization*, ACM Trans. Math. Software, 38 (2011), article 7, doi:10.1145/2049662.2049669.

[50] B. A. Murtagh and M. A. Saunders, *Large-scale linearly constrained optimization*, Math. Program., 14 (1978), pp. 41–72, doi:10.1007/BF01588950.

[51] Y. Nesterov and B. T. Polyak, *Cubic regularization of Newton's method and its global performance*, Math. Program., 108 (2006), pp. 177–205, doi:10.1007/s10107-006-0706-8.

[52] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed., Springer Ser. Oper. Res. Financ. Eng., Springer, New York, 2006, doi:10.1007/978-0-387-40065-5.

[53] M. Weiser, P. Deuflhard, and B. Erdmann, *Affine conjugate adaptive Newton methods for nonlinear elastomechanics*, Optim. Methods Softw., 22 (2007), pp. 413–431, doi:10.1080/10556780600605129.