COMPUTATIONAL & APPLIED MATHEMATICS

Volume 30, N. 1, pp. 19–52, 2011

ISSN 0101-8205
www.scielo.br/cam

Derivative-free methods for nonlinear programming with general lower-level constraints*

M.A. DINIZ-EHRHARDT[1], J.M. MARTÍNEZ[1] and L.G. PEDROSO[2]

[1]Department of Applied Mathematics, IMECC-UNICAMP
University of Campinas, 13083-859 Campinas, SP, Brazil
[2]Department of Mathematics, Federal University of Paraná
81531-980 Curitiba, PR, Brazil

E-mails: {cheti|martinez}@ime.unicamp.br, lucaspedroso@ufpr.br

Abstract. Augmented Lagrangian methods for derivative-free continuous optimization with constraints are introduced in this paper. The algorithms inherit the convergence results obtained by Andreani, Birgin, Martínez and Schuverdt for the case in which analytic derivatives exist and are available. In particular, feasible limit points satisfy KKT conditions under the Constant Positive Linear Dependence (CPLD) constraint qualification. The form of our main algorithm allows us to employ well established derivative-free subalgorithms for solving lower-level constrained subproblems. Numerical experiments are presented.

Mathematical subject classification: 90C30, 90C56.

Key words: nonlinear programming, Augmented Lagrangian, global convergence, optimality conditions, derivative-free optimization, constraint qualifications.

1 Introduction

In many applications one needs to solve finite-dimensional optimization problems in which derivatives of the objective functions or of the constraints are not available. A comprehensive book describing many of these situations has been

#CAM-230/10. Received: 15/VIII/10. Accepted: 05/I/11.

*This work was supported by PRONEX-Optimization (PRONEX – CNPq/ FAPERJ E-26/ 171.510/2006 – APQ1), FAPESP (Grants 2006/53768-0 and 2004/15635-2) and CNPq.

recently published by Conn, Scheinberg and Vicente [9]. In this paper we introduce algorithms for solving constrained minimization problems of this class, in which constraints are divided in two levels, as in [1]. The shifted-penalty approach that characterizes Augmented Lagrangian methods is applied only with respect to the upper-level constraints whereas the lower level constraints are explicitly included in all the subproblems solved at the main algorithm. Lower-level constraints are not restricted to the ones that define boxes or polytopes. In general we will assume that derivatives with respect to lower-level constraints are available, but derivatives with respect to upper-level constraints and objective function are not.

The form of the problems addressed in this paper is the following:

$$\text{Minimize } f(x) \text{ subject to } h(x) = 0, g(x) \leq 0, x \in \Omega. \tag{1}$$

The set Ω represents *lower-level constraints* of the form

$$\underline{h}(x) = 0, \ \underline{g}(x) \leq 0. \tag{2}$$

In the most simple case, Ω will take the form of an n-dimensional box:

$$\Omega = \{x \in \mathbb{R}^n \mid \ell \leq x \leq u\}. \tag{3}$$

We will assume that the functions $f : \mathbb{R}^n \to \mathbb{R}, h : \mathbb{R}^n \to \mathbb{R}^m, g : \mathbb{R}^n \to \mathbb{R}^p, \underline{h} : \mathbb{R}^n \to \mathbb{R}^{\underline{m}}, \underline{g} : \mathbb{R}^n \to \mathbb{R}^{\underline{p}}$ are continuous. First derivatives will be assumed to exist for some convergence results but they will not be used in the algorithmic calculations at all. It is important to stress that the algorithms presented here *do not* rely on derivative discretizations, albeit some theoretical convergence results evoke discrete derivative ideas.

We aim to solve (1) using a derivative-free Augmented Lagrangian approach inspired in [1]. Given $\rho > 0$, $\lambda \in \mathbb{R}^m$, $\mu \in \mathbb{R}^p_+$, $x \in \mathbb{R}^n$, we define the Augmented Lagrangian $L_\rho(x, \lambda, \mu)$ by:

$$L_\rho(x, \lambda, \mu) =$$
$$f(x) + \frac{\rho}{2}\left\{ \sum_{i=1}^{m} \left[h_i(x) + \frac{\lambda_i}{\rho} \right]^2 + \sum_{i=1}^{p} \left[\max\left\{ 0, g_i(x) + \frac{\mu_i}{\rho} \right\} \right]^2 \right\}. \tag{4}$$

At each (outer) iteration the main algorithm will minimize (approximately) $L_\rho(x, \lambda, \mu)$ subject to $x \in \Omega$ using some derivative-free method and, between outer iterations, the Lagrange multipliers λ, μ and the penalty parameter ρ will be conveniently updated. The definition (4) corresponds to the well-known PHR (Powell-Hestenes-Rockafellar) [12, 24, 27] classical Augmented Lagrangian formulation. See, also, [7, 8].

In the *global optimization* framework one finds an approximate global minimizer of $L_\rho(x, \lambda, \mu)$ on Ω at each outer iteration. In the limit, it may be proved that a global minimizer of the original problem up to an arbitrary precision is found [5]. Global minimization subalgorithms are usually expensive, but the Augmented Lagrangian approach exhibits a practical property called "preference for global minimizers" [1], thanks to which one usually finds local minimizers with suitable small objective function values if efficient local minimization algorithms are used for solving the subproblems. In other words, even if we do not find global minimizers of the subproblems, the global results [5] tend to explain why the algorithms work with the sole assumption of continuity of objective function and constraints.

We conjecture that the effectivity of Augmented Lagrangian tools in derivative-free optimization is associated with the main motivation of the general Augmented Lagrangian approach. Augmented Lagrangian methods are not motivated by Newtonian ideas (which involve iterative resolution of linearized problems associated with derivatives). Instead, these methods are based on the idea of minimizing penalized problems with shifted constraints. The most natural updating rule for the multipliers does not use derivatives at all, as its motivation comes from a modified-displacement strategy [24]. Continuity is the only smoothness property of the involved functions that supports the plausibility of Augmented Lagrangian procedures for constrained optimization.

The division of the constraints between upper-level and lower-level ones may be due to different reasons. In our algorithms, we will adopt the point of view that the upper level contains "difficult" constraints whose derivatives are not available and that the lower level contains "easier" constraints whose explicit derivatives could be employed. Derivative-free methods for minimization with (only) lower-level constraints are supposed to exist (see [9, 23]) albeit this as-

sumption is irrelevant from the theoretical point of view. However it is important to put in relief that we could use many known derivative-free algorithms to minimize $L_\rho(x, \lambda, \mu)$ subject to $x \in \Omega$, including directional direct search methods, simplex methods or algorithms based on polynomial interpolation [9]. Sometimes lower-level constraints are easy in the sense that we can deal with them using direct directional search methods. For instance, we can use the MADS algorithm [3, 4], which has been proven to be a very good choice for problems where $\Omega = \overline{Int(\Omega)}$. Although most frequently lower-level constraints define boxes or polytopes, more general situations are not excluded at all. Sometimes lower-level constraints cannot be violated, because they involve fundamental definitions or because the objective function may not be defined when they are not fulfilled.

The choice of the algorithm that should be used to solve the subproblems depends on the nature of the lower-level constraints. The best known case is when Ω is an n-dimensional box. We will develop a special algorithm for this case. The algorithm employs an arbitrary derivative-free box-constraint (or even unconstrained) minimization solver which we complement with a local coordinate search in order to ensure convergence. When Ω is defined by linear (equality or inequality) constraints, the technique of positive generators on cones ([9], Chapter 13) may be employed. See [15, 17, 20].

Assume that $\bar{x}$ is a feasible point of a smooth nonlinear programming problem whose constraints are $\bar{h}_i(x) = 0, i \in I$, $\bar{g}_j(x), j \in J$ and that the active constraints at $\bar{x}$ are (besides the equalities) $\bar{g}_j(x) \leq 0, j \in J_0$. Let $I_1 \subseteq I$, $J_1 \subseteq J_0$. We say that the gradients $\nabla \bar{h}_i(\bar{x})(i \in I_1)$, $\nabla \bar{g}_j(\bar{x})(j \in J_1)$ are *positively linearly dependent* if

$$\sum_{i \in I_1} \lambda_i \nabla \bar{h}_i(\bar{x}) + \sum_{j \in J_1} \mu_j \nabla \bar{g}_j(\bar{x}) = 0,$$

where

$$\lambda_i \in \mathbb{R}\ \forall i \in I_1, \mu_j \geq 0\ \forall j \in J_1 \quad \text{and} \quad \sum_{i \in I_1} |\lambda_i| + \sum_{j \in J_1} \mu_j > 0.$$

The CPLD condition says that, when a subset of gradients of active constraints is positively linearly dependent at $\bar{x}$, then the same gradients remain linearly dependent for all x (feasible or not) in a neighborhood of $\bar{x}$. Therefore, CPLD

is strictly weaker than the Mangasarian-Fromovitz (MFCQ) constraint qualification [21, 28].

The CPLD condition was introduced in [26] and its status as a constraint qualification was elucidated in [2]. In [1] an Augmented Lagrangian algorithm for minimization with arbitrary lower-level constraints was introduced and it was proved that feasible limit points that satisfy CPLD necessarily fulfill the KKT optimality conditions. The derivative-free algorithms introduced in this paper recover the convergence properties of [1].

In [14], Kolda, Lewis and Torczon proposed a derivative-free Augmented Lagrangian algorithm for nonlinear programming problems with a combination of general and linear constraints. In terms of (1), the set Ω is defined by a polytope in [14]. The authors use the approach of [7], by means of which inequality constraints are transformed into equality constraints with the addition of slack non-negative variables. Employing smoothness assumptions, they reproduce the convergence results of [7]. This means that feasible limit points satisfy KKT conditions, provided that the Linear Independence Constraint Qualification (LICQ) holds. In a more recent technical report [19], Lewis and Torczon used the framework of [1] for dealing with linear constraints in the lower level and employed their derivative-free approach for solving the subproblems. The proposed algorithm inherits the theoretical properties of [1], which means that the results in [19] are based on the CPLD constraint qualification. However, the authors don't consider nonlinear constraints in the lower level set, which is possible in our approach.

This paper is organized as follows. In Section 2 we recall the method and theoretical results found in [1]. In Section 3 we introduce a derivative-free version of [1] for the case in which the lower-level set is a box. In Section 4 we introduce a derivative-free version of the Augmented Lagrangian method for arbitrary lower-level constraints. Section 5 describes some numerical results. Finally, we make some comments and present our conclusions about this work in Section 6.

Notation

- The symbol $\|\cdot\|$ denotes an arbitrary vector norm.

- The canonical basis of $\mathbb{R}^n$ will be denoted $\{e^1, \ldots, e^n\}$.
- For all $y \in \mathbb{R}^n$, $y_+ = (\max\{0, y_1\}, \ldots, \max\{0, y_n\})^T$.
- $\mathbb{N} = \{0, 1, \ldots\}$.
- If v is a vector and t is a scalar, the statement $v \leq t$ means that $v_i \leq t$ for all the coordinates i.
- $[a, b]^m = [a, b] \times [a, b] \times \ldots \times [a, b]$ m times.

2 Preliminary results

In this section we recall the main algorithm presented in [1] for solving (1) with Ω defined by (2). We will also mention the convergence theorems that are relevant for our present research.

Algorithm 1 (Derivative-based Augmented Lagrangian)

The parameters that define the algorithm are: $\tau \in [0, 1)$, $\gamma > 1$, $\lambda_{\min} < \lambda_{\max}$, $\mu_{\max} > 0$. At the first outer iteration we use a penalty parameter $\rho_1 > 0$ and safeguarded Lagrange multipliers estimates $\bar{\lambda}^1 \in \mathbb{R}^m$ and $\bar{\mu}^1 \in \mathbb{R}^p$ such that

$$\bar{\lambda}_i^1 \in [\lambda_{\min}, \lambda_{\max}] \; \forall i = 1, \ldots, m \quad \text{and} \quad \mu_i^1 \in [0, \mu_{\max}] \; \forall i = 1, \ldots, p.$$

Finally, $\{\varepsilon_k\}$ is a sequence of positive numbers that satisfies

$$\lim_{k \to \infty} \varepsilon_k = 0.$$

Step 1. *Initialization.*

Set $k \leftarrow 1$.

Step 2. *Solve the subproblem.*

Compute $x^k \in \mathbb{R}^n$ such that there exist $v^k \in \mathbb{R}^{\underline{m}}$, $w^k \in \mathbb{R}^{\underline{p}}$ satisfying

$$\|\nabla L_{\rho_k}(x^k, \bar{\lambda}^k, \bar{\mu}^k) + \sum_{i=1}^{\underline{m}} v_i^k \nabla \underline{h}_i(x^k) + \sum_{i=1}^{\underline{p}} w_i^k \nabla \underline{g}_i(x^k)\| \leq \varepsilon_k, \tag{5}$$

$$w^k \geq 0, \underline{g}(x^k) \leq \varepsilon_k, \tag{6}$$

$$w_i^k = 0 \quad \text{whenever} \quad \underline{g}_i(x^k) < -\varepsilon_k, \ \text{for all } i = 1, \ldots, \underline{p}, \tag{7}$$

$$\|\underline{h}(x^k)\| \leq \varepsilon_k. \tag{8}$$

Step 3. *Estimate multipliers.*

For all $i = 1, \ldots, m$, compute

$$\lambda_i^{k+1} = \bar{\lambda}_i^k + \rho_k h_i(x^k) \tag{9}$$

and

$$\bar{\lambda}_i^{k+1} \in [\lambda_{\min}, \lambda_{\max}]. \tag{10}$$

For all $i = 1, \ldots, p$, compute

$$\mu_i^{k+1} = \max\{0, \bar{\mu}_i^k + \rho_k g_i(x^k)\}, \tag{11}$$

$$V_i^k = \max\left\{g_i(x^k), -\frac{\bar{\mu}_i^k}{\rho_k}\right\},$$

and

$$\bar{\mu}_i^{k+1} \in [0, \mu_{\max}]. \tag{12}$$

Step 4. *Update penalty parameter.*

If $k > 1$ and

$$\max\{\|h(x^k)\|_\infty, \|V^k\|_\infty\} > \tau \max\{\|h(x^{k-1})\|_\infty, \|V^{k-1}\|_\infty\},$$

define

$$\rho_{k+1} = \gamma \rho_k.$$

Else, define

$$\rho_{k+1} = \rho_k.$$

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

Lemma 1 below shows that the points x^k generated by Algorithm 1 are, approximately, KKT points of the problem, which is a consequence of requirements (5–8). Lemma 2 shows that the complementarity conditions respect to constraints $g(x) \leq 0$ are satisfied for k large enough.

Lemma 1. *Assume that* $\{x^k\}$ *is a sequence generated by Algorithm* 1. *Then, for all* $k = 1, 2, \ldots$ *we have:*

$$\left\|\nabla f(x^k) + \nabla h(x^k)\lambda^{k+1} + \nabla g(x^k)\mu^{k+1} + \nabla \underline{h}(x^k)v^k + \nabla \underline{g}(x^k)w^k\right\| \leq \varepsilon_k,$$

where

$$w^k \geq 0, \quad w_i^k = 0 \quad \textit{whenever} \quad \underline{g}_i(x^k) < -\varepsilon_k,$$
$$\underline{g}_i(x^k) \leq \varepsilon_k \;\forall\; i = 1, \ldots, \underline{p}, \;\; \|\underline{h}(x^k)\| \leq \varepsilon_k.$$

Proof. The proof follows from (5–8) using the definitions (9) and (11). □

Lemma 2. *Assume that the sequence* $\{x^k\}$ *is generated by Algorithm* 1 *and that* K *is an infinite sequence of indices such that*

$$\lim_{k \in K} x^k = x^*.$$

Suppose that $g_i(x^*) < 0$. *Then, there exists* $k_0 \in K$ *such that*

$$\mu_i^{k+1} = 0 \quad \textit{for all} \quad k \in K, k \geq k_0.$$

Proof. See the proof of formula (4.10) in Theorem 4.2 of [1]. □

Theorem 1. *Assume that* x^* *is a limit point of a sequence generated by Algorithm* 1. *Then:*

1. *If the sequence of penalty parameters* $\{\rho_k\}$ *is bounded,* x^* *is a feasible point of* (1). *Otherwise, at least one of the following two possibilities holds:*

   - *The point* x^* *satisfies the KKT conditions of the problem*

     $$\textit{Minimize } \|h(x)\|_2^2 + \|g(x)_+\|_2^2 \quad \textit{subject to} \quad \underline{h}(x) = 0, \; \underline{g}(x) \leq 0.$$

   - *The CPLD constraint qualification corresponding to the lower-level constraints* $\underline{h}(x) = 0, \; \underline{g}(x) \leq 0$ *does not hold at* x^*.

2. *If x^* is a feasible point of* (1) *then at least one of the following two possibilities holds:*

   - *The KKT conditions of* (1) *are fulfilled at x^*.*
   - *The CPLD constraint qualification corresponding to all the constraints of* (1) *($h(x) = 0, g(x) \leq 0, \underline{h}(x) = 0, \underline{g}(x) \leq 0$) does not hold at x^*.*

Proof. See Theorems 4.1 and 4.2 of [1]. □

Theorem 1 represents the type of global convergence result that we want to prove for the algorithms presented in this paper. In [1], under additional assumptions, it is proved that the penalty parameters remain bounded when one applies Algorithm 1 to (1). The first part of Theorem 1 says that the algorithm behaves in the best possible way in the process of finding feasible points. This result cannot be improved since the feasible region could be empty and, on the other hand, the algorithm is not equipped for finding global minimizers. It must be observed that, since the lower-level set is generally simple, the CPLD condition related to Ω is usually satisfied at all the lower-level feasible points. The second part of the theorem (which corresponds to Theorem 4.2 of [1]) says that, under the CPLD constraint qualification, every feasible limit point is KKT. Since CPLD is weaker than popular constraint qualifications like LICQ (regularity) or MFCQ (Mangasarian-Fromovitz), this result is stronger than properties that are based on LICQ or MFCQ. The consequence is that, roughly speaking, Algorithm 1 "works" when it generates a bounded sequence (so that limit points necessarily exist). The first requirement for this is, of course, that the conditions (5–8) must be effectively satisfied at every outer iteration, otherwise the algorithm would not be well defined. This requirement must be analyzed in every particular case. A sufficient condition for the boundedness of the sequence is the boundedness of the set

$$\{x \in \mathbb{R}^n \mid \underline{g}(x) \leq \varepsilon, \|\underline{h}(x)\| \leq \varepsilon\} \quad \text{for some} \quad \varepsilon > 0.$$

This condition holds, for example, if the lower-level constraints include box constraints on all the variables.

3 Derivative-free method for box lower-level constraints

In this section we define a derivative-free method that applies to problem (1) when Ω is a box defined by (3).

Algorithm 2 (Derivative-free Augmented Lagrangian for box constraints in the lower level)

Let $\tau \in [0, 1)$, $\gamma > 1$, $\lambda_{\min} < \lambda_{\max}$, $\mu_{\max} > 0$, $\rho_1 > 0$, $\bar{\lambda}^1$, $\bar{\mu}^1$ and $\{\varepsilon_k\}$ be as in Algorithm 1. Steps 1, 3 and 4 of Algorithm 2 are identical to those of Algorithm 1. Step 2 is replaced by the following:

Step 2. *Solve the subproblem.*

Choose a positive tolerance $\delta_k < (u_i - \ell_i)/2$, $i = 1, \ldots, n$, such that

$$\delta_k \leq \min\left\{\frac{\varepsilon_k}{\rho_k}, \varepsilon_k\right\}. \tag{13}$$

Find $x^k \in \Omega$ such that, for all $i = 1, \ldots, n$,

$$x^k \pm \delta_k e^i \in \Omega \Rightarrow L_{\rho_k}(x^k \pm \delta_k e^i, \bar{\lambda}^k, \bar{\mu}^k) \geq L_{\rho_k}(x^k, \bar{\lambda}^k, \bar{\mu}^k). \tag{14}$$

Since the algorithm found in [19] handles linear constrained problems, it could also be applied to a box constrained problem. The main difference between this algorithm and Algorithm 2 relies on the choice of the solver for the subproblem. We encourage the use of any derivative-free technique able to find points satisfying (14), whereas in [19] the authors use Generating Set Search methods for solving the subproblems, taking advantage of the geometry of the linear constraints.

Theorem 2. *Assume that $\nabla f, \nabla h, \nabla g$ satisfy Lipschitz conditions on the box Ω. Then Algorithm 2 is a particular case of Algorithm 1. Moreover:*

- *The sequence $\{x^k\}$ is well defined for all k and admits limit points.*
- *If the sequence of penalty parameters $\{\rho_k\}$ is bounded, x^* is feasible. Otherwise, every limit point x^* is a stationary point of the box constrained problem*

$$\textit{Minimize } \|h(x)\|_2^2 + \|g(x)_+\|_2^2 \textit{ subject to } \quad x \in \Omega.$$

- *If x^* is a feasible limit point, then at least one of the following possibilities holds:*

  1. *The point x^* fulfills the KKT conditions of*

  $$\textit{Minimize } f(x), \textit{ subject to } h(x) = 0, g(x) \leq 0, x \in \Omega.$$

  2. *The CPLD constraint qualification is not satisfied at x^* for the constraints $h(x) = 0, g(x) \leq 0, x \in \Omega$.*

Proof. By (4), (10), (12) and the Lipschitz assumption, there exists $M > 0$ such that for all $x, y \in \Omega$ we have:

$$\|\nabla L_{\rho_k}(x, \bar{\lambda}^k, \bar{\mu}^k) - \nabla L_{\rho_k}(y, \bar{\lambda}^k, \bar{\mu}^k)\|_\infty \leq M\rho_k \|x - y\|_\infty. \tag{15}$$

Assume that x^k is computed at Step 2 of Algorithm 2. Given $i \in \{1, \ldots, n\}$, since $\delta_k < (u_i - \ell_i)/2$, we may consider three possible cases:

- $\ell_i \leq x_i^k - \delta_k$ and $x_i^k + \delta_k \leq u_i$;
- $\ell_i > x_i^k - \delta_k$ and $x_i^k + \delta_k \leq u_i$;
- $\ell_i \leq x_i^k - \delta_k$ and $x_i^k + \delta_k > u_i$.

Consider the first case. By (14) and the Mean Value Theorem, there exists $\theta_i^k \in [-\delta_k, \delta_k]$ such that

$$\left[\nabla L_{\rho_k}(x^k + \theta_i^k e^i, \bar{\lambda}^k, \bar{\mu}^k)\right]_i = 0.$$

Then by (15)

$$\left|[\nabla L_{\rho_k}(x^k, \bar{\lambda}^k, \bar{\mu}^k)]_i\right| \leq M\rho_k\delta_k. \tag{16}$$

Now consider the second case. By (14) and the Mean Value Theorem, there exists $\theta_i^k \in [0, \delta_k]$ such that

$$\left[\nabla L_{\rho_k}(x^k + \theta_i^k e^i, \bar{\lambda}^k, \bar{\mu}^k)\right]_i \geq 0.$$

So there exists $w_i^k \geq 0$ such that

$$\left[\nabla L_{\rho_k}(x^k + \theta_i^k e^i, \bar{\lambda}^k, \bar{\mu}^k)\right]_i - w_i^k = 0.$$

Therefore, by (15),

$$\left|[\nabla L_{\rho_k}(x^k, \bar{\lambda}^k, \bar{\mu}^k)]_i - w_i^k\right| \leq M\rho_k\delta_k. \tag{17}$$

Analogously, in the third case we obtain that there exists $w_i^k \geq 0$ such that

$$\left|[\nabla L_{\rho_k}(x^k, \bar{\lambda}^k, \bar{\mu}^k)]_i + w_i^k\right| \leq M\rho_k\delta_k. \tag{18}$$

From (16), (17) and (18), we deduce that there exists $w^k \geq 0$ such that

$$\left\|\nabla L_{\rho_k}(x^k, \bar{\lambda}^k, \bar{\mu}^k) + \sum_{i=1}^{n} \pm w_i^k e^i\right\|_\infty \leq M\rho_k\delta_k, \tag{19}$$

where

$$w_i^k = 0 \text{ if } x_i^k - \ell_i \geq \delta_k \text{ and } x_i^k - u_i \leq -\delta_k.$$

The sign $-$ takes place in (19) if

$$x_i^k - \ell_i < \delta_k \text{ and } x_i^k \leq u_i - \delta_k,$$

while the sign $+$ takes place in (19) if

$$x_i^k - \ell_i \geq \delta_k \text{ and } x_i^k - u_i > -\delta_k.$$

Then, by (13), x^k satisfies (5–8) if one redefines $\varepsilon_k \leftarrow \max\{\varepsilon_k, M\varepsilon_k\}$.

To complete the proof of the theorem, observe first that it is always possible to satisfy (14) defining a δ_k-grid on Ω and taking x^k as a global minimizer of $L_{\rho_k}(x, \bar{\lambda}^k, \bar{\mu}^k)$ on that grid. Since Ω is bounded, the sequence is in a compact set and limit points exist. Moreover, the box constraints that define Ω satisfy CPLD, therefore, by Theorem 1, limit points are stationary points of $\|h(x)\|_2^2 + \|g(x)_+\|_2^2$. The last part of the thesis is a direct consequence of Theorem 1. □

4 Derivative-free method for arbitrary lower-level constraints

In this section we define a derivative-free method of Augmented Lagrangian type for solving (1) with arbitrary lower-level constraints. Recall that in Section 3 the case in which the lower-level constraints define a box has been addressed.

In order to minimize the Augmented Lagrangian subject to the lower-level constraints, we need to use an "admissible" derivative-free algorithm, having appropriate theoretical properties. Roughly speaking, when objective function and constraints are differentiable, an admissible algorithm should compute KKT points of the subproblem up to any required precision.

Let $F : \mathbb{R}^n \to \mathbb{R}$, $\underline{h} : \mathbb{R}^n \to \mathbb{R}^{\underline{m}}$, $\underline{g} : \mathbb{R}^n \to \mathbb{R}^{\underline{p}}$ be continuously differentiable. We say that an iterative algorithm is *admissible* with respect to the problem

$$\text{Minimize } F(x) \text{ subject to } \underline{h}(x) = 0, \underline{g}(x) \leq 0 \tag{20}$$

if it does not make use of analytical derivatives of F and, independently of x^0, it computes a sequence $\{x^\nu\}$ with the following properties:

- $\{x^\nu\}$ is bounded;
- Given $\varepsilon > 0$ and $\nu_1 \in \mathbb{N}$, there exist $\nu \geq \nu_1$, $x^\nu \in \mathbb{R}^n$, $v^\nu \in \mathbb{R}^{\underline{m}}$, $w^\nu \in \mathbb{R}^{\underline{p}}_+$, satisfying:

$$\|\nabla F(x^\nu) + \sum_{i=1}^{\underline{m}} v_i^\nu \nabla \underline{h}_i(x^\nu) + \sum_{i=1}^{\underline{p}} w_i^\nu \nabla \underline{g}_i(x^\nu)\| \leq \varepsilon, \tag{21}$$

$$w^\nu \geq 0, \underline{g}(x^\nu) \leq \varepsilon, \tag{22}$$

$$w_i^\nu = 0 \quad \text{whenever} \quad \underline{g}_i(x^\nu) < -\varepsilon, \ \forall\, i = 1, \ldots, \underline{p}, \tag{23}$$

$$\|\underline{h}(x^\nu)\| \leq \varepsilon. \tag{24}$$

In this case, we say that x^ν is an ε-KKT point of the problem (20).

When we talk about lower-level constraints, we have in mind not only boxes and polytopes, as [14, 15], but also more general constraints that can be handled using, for example, the extreme barrier approach (see [9], Chapter 13). Barrier methods can be useful for solving subproblems when easy inequality constraints define feasible sets Ω such that

$$\Omega = \overline{Int(\Omega)}.$$

In these cases, incorporating the constraints which define Ω in the upper-level set may be inconvenient. A reasonable algorithm for solving the problem of minimizing a function subject to this type of constraints can be found in [3, 4].

We will assume that $G_\rho(x, \lambda, \mu, \delta)$ approximates the gradient of the Augmented Lagrangian, in the sense that there exists $M > 0$ such that, for all

$$x \in \mathbb{R}^n,\ \lambda \in [\lambda_{\min}, \lambda_{\max}]^m,\ \mu \in [0, \mu_{\max}]^p,\ \delta > 0,$$

$$\|G_\rho(x, \lambda, \mu, \delta) - \nabla L_\rho(x, \lambda, \mu)\| \leq \delta M \rho. \tag{25}$$

Many possible definitions of G can be given satisfying (25) if $\nabla f, \nabla h, \nabla g$ satisfy Lipschitz conditions. An appealing possibility is to define G as a simplex gradient [9]. It may not be necessary to use many auxiliary points to compute G because this approximation can be built using previous iterates of the subalgorithm.

Algorithm 3, defined below, applies to the general problem (1). For this algorithm we will prove that the convergence properties of Algorithm 1 can be recovered under suitable assumptions on the optimization problem. For the application of Algorithm 3 we will assume that the gradients of the lower-level constraints $\nabla \underline{h}$ and $\nabla \underline{g}$ are available.

For each outer iteration k we will denote by $\{x^{k,v}\}$ a sequence generated by an admissible algorithm applied to the problem

$$\text{Minimize } L_{\rho_k}(x, \bar{\lambda}^k, \bar{\mu}^k) \text{ subject to } \underline{h}(x) = 0, \underline{g}(x) \leq 0. \tag{26}$$

We assume that the computational form of the admissible algorithm includes some internal stopping criterion that is going to be tested for each v. When this stopping criterion is fulfilled we check the approximate fulfillment of (21–24) without using analytic derivatives of f, h, g. Of course, some admissible algorithms may ensure that the approximate KKT conditions are satisfied when their convergence criteria are met, in which case it is not necessary to check the approximate KKT conditions.

Algorithm 3 (Derivative-free Augmented Lagrangian for general constraints in the lower level)

Let $\tau \in [0, 1)$, $\gamma > 1$, $\lambda_{\min} < \lambda_{\max}$, $\mu_{\max} > 0$, $\rho_1 > 0$, $\bar{\lambda}^1$, $\bar{\mu}^1$ and $\{\varepsilon_k\}$ be as in Algorithm 1. Steps 1, 3 and 4 of Algorithm 3 are identical to those of Algorithm 1. Step 2 is replaced by the following:

Step 2. *Solve the subproblem.*

Choose a tolerance δ_k such that

$$\delta_k \leq \frac{\varepsilon_k}{\rho_k}. \tag{27}$$

Let $\{\delta_{k,\nu}\}$ be such that $\delta_{k,\nu} \in (0, \delta_k]$ for all ν and

$$\lim_{\nu \to \infty} \delta_{k,\nu} = 0.$$

Step 2.1. Set $\nu \leftarrow 1$.

Step 2.2. Compute $x^{k,\nu}$.

Step 2.3. If $x^{k,\nu}$ does not satisfy the intrinsic stopping criterion of the admissible subalgorithm, set $\nu \leftarrow \nu + 1$ and go to Step 2.2.

Step 2.4. If $\|\underline{h}(x^{k,\nu})\| > \varepsilon_k$ or there exists $i \in \{1, \ldots, \underline{p}\}$ such that $\underline{g}_i(x^{k,\nu}) > \varepsilon_k$, set $\nu \leftarrow \nu + 1$ and go to Step 2.2.

Step 2.5. Compute $G_{\rho_k}(x^{k,\nu}, \bar{\lambda}^k, \bar{\mu}^k, \delta_{k,\nu})$, $\nabla \underline{h}(x^{k,\nu})$, $\nabla \underline{g}(x^{k,\nu})$.

Step 2.6. Define

$$I_\nu = \{i \in \{1, \ldots, \underline{p}\} \mid \underline{g}_i(x^{k,\nu}) < -\varepsilon_k\}.$$

Solve, with respect to (v, w), the following problem:

$$\text{Minimize} \quad \left\| G_{\rho_k}(x^{k,\nu}, \bar{\lambda}^k, \bar{\mu}^k, \delta_{k,\nu}) + \sum_{i=1}^{\underline{m}} v_i \nabla \underline{h}_i(x^{k,\nu}) + \sum_{i=1}^{\underline{p}} w_i \nabla \underline{g}_i(x^{k,\nu}) \right\| \tag{28}$$

$$\text{subject to} \quad w \geq 0 \quad \text{and} \quad w_i = 0 \text{ if } i \in I_\nu.$$

Step 2.7. If, at a solution (v, w) of (28) we have that

$$\left\| G_{\rho_k}(x^{k,\nu}, \bar{\lambda}^k, \bar{\mu}^k, \delta_{k,\nu}) + \sum_{i=1}^{\underline{m}} v_i \nabla \underline{h}_i(x^{k,\nu}) + \sum_{i=1}^{\underline{p}} w_i \nabla \underline{g}_i(x^{k,\nu}) \right\| \leq \varepsilon_k, \tag{29}$$

define $\nu^*(k) = \nu$, $x^k = x^{k,\nu^*(k)}$, $v^k = v$, $w^k = w$ and go to Step 3. Else, set $\nu \leftarrow \nu + 1$ and go to Step 2.2.

The purpose of Step 2 above is to find a point $x^{k,\nu^*(k)}$ and multipliers v^k, w^k that verify (5-8), but with the actual gradient of ∇L_{ρ_k} replaced by the simplex gradient G_{ρ_k}. At (28) we compute approximate Lagrange multipliers v, w with respect to lower-level constraints. In some cases, the basic admissible algorithm may provide these multipliers so that the step (28) may not be necessary. If $\|\cdot\|$ is the Euclidean norm, (28) involves the minimization of a convex quadratic with non-negative constraints. So, its resolution is affordable in the context of derivative-free optimization, where functions evaluations are generally very expensive, since neither f, g nor h are computed at (28).

In the following theorem we prove that Algorithm 3 is well defined. This amounts to show that the loop defined by Steps 2.1–2.7 necessarily finishes for some finite ν.

Theorem 3. *Assume that, for all $k = 1, 2, \ldots$ an admissible algorithm with respect to* (26) *is available. Assume, moreover, that the first derivatives of f, h, g, $\underline{h}$, $\underline{g}$ exist and are Lipschitz on a sufficiently large set. Then, Algorithm* 3 *is well defined.*

Proof. For fixed k, assume that the admissible algorithm, whose existence is postulated in the hypothesis, computes a sequence $\{x^{k,\nu}\}$. Since f, h, g have Lipschitz gradients, there exists a Lipschitz constant $\rho_k M$ for the function $\nabla L_{\rho_k}(x^{k,\nu}, \bar{\lambda}^k, \bar{\mu}^k)$ that is valid for all x in a sufficiently large ball that contains the whole sequence and (25) is satisfied. Let ν_1 be such that

$$\rho_k M \delta_{k,\nu} < \varepsilon_k/2 \tag{30}$$

for all $\nu \geq \nu_1$ (note that it is not necessary to know the value of M at all). By the definition of admissible algorithm, there exists $\nu \geq \nu_1$, $v^{k,\nu} \in \mathbb{R}^{\underline{m}}$, $w^{k,\nu} \in \mathbb{R}^{\underline{p}}_+$ such that:

$$\left\| \nabla L_{\rho_k}(x^{k,\nu}, \bar{\lambda}^k, \bar{\mu}^k) + \sum_{i=1}^{\underline{m}} v_i^{k,\nu} \nabla \underline{h}_i(x^{k,\nu}) + \sum_{i=1}^{\underline{p}} w_i^{k,\nu} \nabla \underline{g}_i(x^{k,\nu}) \right\| \leq \varepsilon_k/2, \tag{31}$$

$$w^{k,\nu} \geq 0, \underline{g}(x^{k,\nu}) \leq \varepsilon_k/2, \tag{32}$$

$$\underline{g}_i(x^{k,\nu}) < -\varepsilon_k/2 \Rightarrow w_i^{k,\nu} = 0 \text{ for all } i = 1, \ldots, \underline{p}, \tag{33}$$

$$\|\underline{h}(x^{k,\nu})\| \leq \varepsilon_k/2. \tag{34}$$

Now, by (25) and (30),

$$\|G_{\rho_k}(x^{k,\nu}, \bar{\lambda}^k, \bar{\mu}^k) - \nabla L_{\rho_k}(x^{k,\nu}, \bar{\lambda}^k, \bar{\mu}^k, \delta_{k,\nu})\| \leq \varepsilon_k/2.$$

Thus, by (31),

$$\left\| G_{\rho_k}(x^{k,\nu}, \bar{\lambda}^k, \bar{\mu}^k, \delta_{k,\nu}) + \sum_{i=1}^{\underline{m}} v_i^{k,\nu} \nabla \underline{h}_i(x^k) + \sum_{i=1}^{\underline{p}} w_i^{k,\nu} \nabla \underline{g}_i(x^k) \right\| \leq \varepsilon_k,$$

therefore, by (32–34), the solution of problem (28) necessarily satisfies (29). This means that the loop at Step 2 of Algorithm 3 necessarily finishes in finite time, so the algorithm is well defined. □

Theorem 4. *In addition to the hypotheses of Theorem* 3*, assume that there exists* $\varepsilon > 0$ *such that the set defined by* $\|\underline{h}(x)\| \leq \varepsilon, \underline{g}(x) \leq \varepsilon$ *is bounded. Then, the sequence* $\{x^k\}$ *defined by Algorithm* 3 *is well defined and bounded. Moreover, if* x^* *is a limit point of this sequence, we have:*

1. *The lower-level constraints are satisfied at* x^* *(*$\underline{h}(x^*) = 0, \underline{g}(x^*) \leq 0$*).*
2. *If the sequence of penalty parameters* $\{\rho_k\}$ *is bounded,* x^* *is a feasible point of* (1)*. Otherwise, at least one of the following two possibilities holds:*
   - *The point* x^* *satisfies the KKT conditions of the problem*

     $$\text{Minimize } \|h(x)\|_2^2 + \|g(x)_+\|_2^2 \text{ subject to } \underline{h}(x) = 0, \underline{g}(x) \leq 0.$$

   - *The CPLD constraint qualification corresponding to the lower-level constraints* $\underline{h}(x) = 0, \underline{g}(x) \leq 0$ *does not hold at* x^**.*
3. *If* x^* *is a feasible point of* (1) *then at least one of the following possibilities hold:*

- *The KKT conditions of* (1) *are fulfilled at* x^*.
- *The CPLD constraint qualification corresponding to all the constraints of* (1) *(*$h(x) = 0$, $g(x) \leq 0$, $\underline{h}(x) = 0$, $\underline{g}(x) \leq 0$*) does not hold at* x^*.

Proof. By Theorem 3, the sequence $\{x^k\}$ is well defined. Since $\varepsilon_k \leq \varepsilon$ for k large enough, all the iterates of the method belong to a compact set from some iteration on. This implies that the whole sequence is bounded, so limit points exist.

By (25) we have that:

$$\|G_{\rho_k}(x^k, \bar{\lambda}^k, \bar{\mu}^k, \delta_{k,\nu^*(k)}) - \nabla L_{\rho_k}(x^k, \bar{\lambda}^k, \bar{\mu}^k)\| \leq M\rho_k\delta_{k,\nu^*(k)}.$$

Therefore, since, by (27), $\delta_{k,\nu^*(k)} \leq \delta_k$ and $\rho_k\delta_k \leq \varepsilon_k$,

$$\|G_{\rho_k}(x^k, \bar{\lambda}^k, \bar{\mu}^k, \delta_{k,\nu^*(k)}) - \nabla L_{\rho_k}(x^k, \bar{\lambda}^k, \bar{\mu}^k)\| \leq M\varepsilon_k.$$

Then, by (29),

$$\left\| \nabla L_{\rho_k}(x^k, \bar{\lambda}^k, \bar{\mu}^k) + \sum_{i=1}^{\underline{m}} v_i^k \nabla \underline{h}_i(x^{k,\nu}) + \sum_{i=1}^{\underline{p}} w_i^k \nabla \underline{g}_i(x^{k,\nu}) \right\| \leq (1 + M)\varepsilon_k.$$

Moreover, by Step 2 of Algorithm 3,

$$\|\underline{h}(x^k)\| \leq \varepsilon_k, \underline{g}(x^k) \leq \varepsilon_k$$

and

$$w_i^k = 0 \text{ whenever } \underline{g}_i(x^k) < -\varepsilon_k.$$

Therefore, redefining $\varepsilon_k \leftarrow (1 + M)\varepsilon_k$, we have that Algorithm 3 can be seen as a particular case of Algorithm 1. So the thesis of Theorem 1 holds for Algorithm 3 and the theorem is proved. □

In Theorem 4 we proved that the only requirement for Algorithm 3 to be a satisfactory algorithm for solving (1) with arbitrary lower level constraints is the availability of an admissible algorithm for solving the subproblems. Now we are going to show that, for many reasonable lower-level constraints, Algorithm 2

may be such admissible algorithm. As we mentioned before, reasonable admissible algorithms has been already introduced by several authors for particular classes of lower-level sets. Here we wish to show that, as a last resource, even Algorithm 2 may also be used as admissible algorithm for solving subproblems.

The special case of lower-level constraints that we wish to consider is almost as general as it could be. Essentially, we are going to consider arbitrary lower level constraints with bounds on the variables. More precisely, we define:

$$\Omega = \{x \in \mathbb{R}^n \mid \underline{h}(x) = 0, \underline{g}(x) \leq 0, \ell \leq x \leq u\}, \tag{35}$$

where $\underline{h}$ and $\underline{g}$ are as defined in the Introduction. A very important particular case of (35) is when Ω is described by linear (equality and inequality) constraints and bounds.

For proving the main admissibility result we will need the following assumption.

Assumption A.

If x is a stationary (KKT) point of the problem

$$\textit{Minimize } \|\underline{h}(x)\|_2^2 + \|\underline{g}(x)_+\|_2^2 \textit{ subject to } \ell \leq x \leq u$$

then

$$\underline{h}(x) = 0 \textit{ and } \underline{g}(x) \leq 0.$$

Theorem 5. *Assume that $F\colon \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable and suppose that $\underline{h}$, $\underline{g}$, ℓ, u satisfy Assumption* A. *Then Algorithm* 2 *is an admissible algorithm for the problem*

$$\textit{Minimize } F(x) \textit{ subject to } \underline{h}(x) = 0, \underline{g}(x) \leq 0, \ell \leq x \leq u.$$

Proof. Assume that we apply Algorithm 2 to the problem above. Therefore, F plays the role of f, $\underline{h}$ and $\underline{g}$ stand for h and g, respectively.

First observe that, as required by the definition of admissibility, the generated sequence is well defined and bounded.

Without loss of generality, let us identify $\{x^\nu\}$ as a convergent subsequence of the sequence generated by Algorithm 2. By Theorem 2, Algorithm 2 is a particular case of Algorithm 1. This implies that, for a given $\varepsilon > 0$ and ν large enough, condition (21) necessarily holds.

Also by Theorem 2, every limit point is a stationary point of the infeasibility measure subject to the bounds. Therefore, by Assumption A, (22) and (24) also hold for ν large enough.

Finally, let us prove that (23) also takes place for ν large enough. Suppose, by contradiction, that this is not true. Therefore, there exists $\bar{\varepsilon} > 0$ such that, for all ν large enough, there exits $i = i(\nu)$ satisfying

$$\underline{g}_i(x^\nu) < -\bar{\varepsilon} \text{ and } w_i^\nu > 0.$$

Without loss of generality, since the number of different indices of i is finite, we may assume that $i = i(\nu)$ for all ν. Hence, in the limit, one has $\underline{g}_i(x^*) < 0$. Then, by Lemma 2, $w_i^\nu = 0$ for ν large enough. This is a contradiction, which ends the proof of the theorem. □

Theorem 5 shows that Algorithm 2 is an admissible algorithm. Hence, by Theorem 3, it is possible to use Algorithm 2 to generate a sequence $\{x^{k,\nu}\}$ that can be employed in Step 2 of Algorithm 3, when one is attempting to solve (1) with Ω as in (35).

5 Numerical experiments

In this section, we present some computational results obtained with a Fortran 77 implementation of Algorithm 2. Since there is freedom on the choice of the algorithm for the box constrained subproblems, we tested three derivative-free solvers: Coordinate Search [16], the BOBYQA software [25], based on quadratic approximations and trust region techniques, and the well known Nelder-Mead algorithm [22]. In order to satisfy the condition (14), the points returned by BOBYQA or Nelder-Mead are taken as knots of meshes with density δ_k and, if necessary, other points on these meshes are visited. Eventually, a point satisfying (14) is found. This means that BOBYQA and Nelder-Mead may be followed by some iterations of Coordinate Search. On the other hand, Coordinate Search satisfies (14) naturally, if we set δ_k as the step of the search

on the exit of the algorithm. When using the Nelder-Mead algorithm for the subproblems, we force $f(x) = \infty$ for $x \notin \Omega$, to make sure that all generated points lie within the box. We will call the Augmented Lagrangian with BOBYQA, Nelder-Mead and Coordinate Search as subsolvers, respectively, as AL-BOBYQA, AL-NMead and AL-CSearch.

We say that the Augmented Lagrangian found a solution of a problem if (14) is achieved with tolerance δ_{opt} at a point $\overline{x}$ that is sufficiently feasible, in the sense that, for a tolerance ε_{feas}, $\max\{\|h(\overline{x})\|, \|V(\overline{x})\|\} \leq \varepsilon_{feas}$. We consider that the algorithm fails in the attempt of solving the problem in the following cases:

Failure 1: when it cannot find the solution in up to 50 outer iterations,

Failure 2: when it performs 9 outer iterations without improving the feasibility,

Failure 3: when it evaluates the Augmented Lagrangian function more than 10^6 times in a single call of the subsolver.

5.1 Hock-Schittkowski problems

The first set of problems we considered was the Hock-Schittkowski collection [13], which is composed by 119 differentiable problems. We solved only the 47 problems that have general and box constraints simultaneously. The dimension of the problems varies between 2 and 16, while the number of constraints are between 1 and 38, exceeding 10 in only 5 cases. We used $\delta_{opt} = \varepsilon_{feas} = 10^{-5}$. Table 1 shows the performance of each algorithm for all problems, with respect to the objective function on the solution, $f(x^*)$, and the number of function evaluations, $feval$, while Table 2 lists the feasibility reached in each case. It is worth noticing that it was required as a stopping criterion for the Augmented Lagrangian algorithm that $\max\{\|h(\overline{x})\|, \|V(\overline{x})\|\} \leq 10^{-5}$, and for this reason, in all cases where the algorithm succeeded the measure of feasibility is small. But on the other hand, in the cases where the feasibility is substantially small than the tolerance, we have the impression that the respective algorithm is not having problem on achieving or maintaining the feasibility.

The first line of Table 3 shows the percentage of problems that took 10 to 100 function evaluations to be solved, the second line report those that used 100

to 1000 function evaluations and so on. The last three lines show the amount of failures.

Table 4 shows which method has the best performance for each problem. The first line shows the percentage of problems for which each method evaluated less times the Augmented Lagrangian function, the second and third lines consider the occasions where each method was the second best and worst, respectively, while the last line informs the total amount of failures.

We can see from the tables that BOBYQA was the most economic subproblem solver. On the other hand, even with the greatest number of function evaluations in many cases, AL-NMead was the most robust. Recall that we do not used the "pure" Nelder-Mead method as a subsolver, but the Nelder-Mead method plus Coordinate Search; the same comment can be made for BOBYQA. AL-CSearch exhibited an intermediate performance when we consider computational effort, but was the least robust among the three algorithms.

5.2 Minimum volume problems

The volume of complex regions defined by inequalities in $\mathbb{R}^d$ is very hard to compute. A popular technique, when one needs to compute a volume approximation, consists of using a Monte Carlo approach [6, 11]. We will use some small-dimensional examples ($d = 2$) in order to illustrate the behavior of derivative-free methods in problems in which volumes need to be optimized. In the following examples, one needs to minimize the Monte Carlo approximation of the area of a figure subject to constraints.

Suppose that we want to find the minimal Monte Carlo approximation area of the figure defined by the intersection between two circles that contains n_p given points. With this purpose, we draw a tight rectangle containing the intersection of the circles. Then, a large number of points is generated with uniform distribution inside the rectangle. The area of the intersection is, approximately, the area of the rectangle multiplied by the ratio of random points that dropped inside the intersection of the circles. This is the objective function, the simulated area, that has to be minimized. The constraints arise from the fact that the n_p points have to be inside the intersection. It is worth noticing that it is a different rectangle, which contains the intersection, at each iteration. Our intention is to achieve

	$f(x^*)$			$feval$		
problem	AL-CSearch	AL-BOBYQA	AL-NMead	AL-CSearch	AL-BOBYQA	AL-Nmead
18	5.00000E+00	5.00000E+00	5.00000E+00	805	288	302
19	-6.96200E+03	-6.96182E+03	-6.96181E+03	failure 1	2808	1535
21	-9.99600E+01	-9.99600E+01	-9.99600E+01	57	60	177
23	2.00000E+00	1.99998E+00	2.00001E+00	613	504	841
30	1.00000E+00	1.00000E+00	1.00000E+00	95	81	338
31	6.00003E+00	6.00000E+00	6.00000E+00	506	277	427
34	-8.34030E-01	-8.34033E-01	-8.34029E-01	8505	850	1640
36	-3.30000E+03	-3.30000E+03	-3.30000E+03	235	181	502354
37	-3.45600E+03	-3.45600E+03	-3.45600E+03	2790	888	1422
41	2.00000E+00	1.92593E+00	1.92593E+00	75	598	3125
53	4.09293E+00	4.09299E+00	4.09300E+00	failure 1	2391	4063
54	-3.86222E-01	-3.86222E-01	-3.86222E-01	failure 3	failure 3	failure 3
59	-6.74951E+00	-6.74951E+00	-6.74951E+00	223	114	196
60	3.25682E-02	3.25682E-02	3.25683E-02	571	508	1310
62	-2.62725E+04	-2.62726E+04	-2.62725E+04	2412	2086	503494
65	9.53548E-01	9.53529E-01	9.53529E-01	9962	976	512
66	5.18214E-01	5.18163E-01	5.18163E-01	11783	5609	831
67	-1.16210E+03	-1.16210E+03	-1.16210E+03	4473	848	965
68	-9.14964E-01	-9.15545E-01	-9.15117E-01	21540	1280	27083
69	-9.38575E+02	-9.38578E+02	-9.38578E+02	3160	1119	5383
70	8.21625E-02	7.49846E-03	7.49846E-03	5424	656	455
71	1.70140E+01	1.70140E+01	1.70140E+01	10591	1082	10650
72	7.27384E+02	7.27384E+02	7.27384E+02	8061	26804	6720
74	5.12644E+00	5.12645E+00	5.12648E+00	3827	2527	4345
75	5.17452E+00	5.17434E+00	5.17438E+00	failure 1	1722	2952
80	1.00000E+00	5.39497E-02	5.39497E-02	failure 1	1871	1711
81	5.40176E-02	5.39498E-02	5.39499E-02	23536	3880	3421
83	-3.06655E+04	-3.06655E+04	-3.06655E+04	1777	1311	7752
84	-5.08213E+06	-5.22532E+06	-5.46857E+06	678500	607443	4152499
85	-1.90516E+00	-1.90582E+00	-1.90515E+00	427470	failure 3	60009
87	4.20900E+04	8.99688E+03	4.20900E+04	failure 3	54093	failure 3
95	1.56342E-02	1.56195E-02	1.63730E-02	215	377	1936
96	1.56342E-02	1.56195E-02	1.63730E-02	223	377	1940
97	3.13640E+00	3.13581E+00	4.53333E+00	1754	700	7234029
98	3.13640E+00	3.13581E+00	4.53333E+00	1754	700	7234029
99	-8.24986E+08	-8.30239E+08	-8.31080E+08	failure 1	failure 2	20628
101	1.79162E+03	1.79161E+03	1.79252E+03	215385	202679	144597
102	8.80405E+02	1.14917E+03	8.80046E+02	23618	267740	1068121
103	5.34643E+02	1.45767E+03	5.33738E+02	54425	222196	1820078
104	3.95118E+00	3.95115E+00	3.95115E+00	87385	19614	7993
105	1.13631E+03	1.13631E+03	1.13631E+03	6891	6193	12149
106	7.04925E+03	7.04922E+03	7.04922E+03	657806	37640	504366
111	-4.77324E+01	-4.77103E+01	-4.77612E+01	28630	23292	14553
114	-8.72387E+02	-8.72387E+02	-8.72387E+02	failure 3	failure 3	failure 3
116	1.00575E+00	1.16216E+00	9.88126E-01	136210	99678	6215728
118	6.64820E+02	6.64820E+02	6.64820E+02	220633	3633	186851
119	2.44900E+02	2.44899E+02	2.44901E+02	27407	11234	70403

Table 1 – Results for the Hock-Schitkowski problems.

	$\max\{\|h(\overline{x})\|, \|V(\overline{x})\|\}$		
problem	AL-Csearch	AL-BOBYQA	AL-Nmead
18	6.93835E-06	4.22791E-08	3.41580E-06
19	8.61303E-05	1.48703E-06	8.90075E-07
21	0.00000E+00	0.00000E+00	0.00000E+00
23	0.00000E+00	4.20926E-06	9.06558E-06
30	0.00000E+00	0.00000E+00	0.00000E+00
31	5.60586E-06	4.10645E-08	5.86759E-07
34	1.07579E-06	2.64828E-07	8.58953E-06
36	0.00000E+00	1.27542E-12	9.53323E-07
37	0.00000E+00	4.22483E-06	4.38670E-06
41	0.00000E+00	7.85883E-06	9.24319E-06
53	1.52588E-05	3.54130E-06	3.69152E-06
54	5.60000E+03	5.60000E+03	5.60000E+03
59	0.00000E+00	0.00000E+00	0.00000E+00
60	4.34169E-07	1.06845E-06	5.92753E-06
62	1.11022E-16	9.76119E-06	4.65999E-06
65	0.00000E+00	9.43265E-08	1.41357E-07
66	4.15393E-06	6.11729E-07	8.58920E-07
67	9.53939E-06	2.93764E-07	2.17418E-07
68	3.06824E-06	1.05255E-07	1.91330E-06
69	1.38778E-17	4.27808E-06	5.43593E-06
70	0.00000E+00	0.00000E+00	0.00000E+00
71	9.08319E-06	8.04100E-07	6.96594E-06
72	7.29974E-06	7.29978E-06	7.29972E-06
74	9.52878E-06	4.90520E-06	1.77604E-06
75	1.15089E-05	9.18479E-06	3.91491E-06
80	7.39619E-05	2.41344E-06	3.56675E-06
81	6.31320E-06	4.08971E-06	1.77111E-06
83	8.76166E-06	9.02256E-06	8.16030E-06
84	0.00000E+00	0.00000E+00	0.00000E+00
85	2.42318E-06	2.57279E-03	5.87496E-06

Table 2 – Feasibility for the Hock-Schitkowski problems.

	$\max\{\|h(\overline{x})\|, \|V(\overline{x})\|\}$		
problem	AL-Csearch	AL-BOBYQA	AL-Nmead
87	5.76242E+02	7.64699E-06	5.76242E+02
95	0.00000E+00	4.12168E-06	0.00000E+00
96	0.00000E+00	4.12168E-06	0.00000E+00
97	0.00000E+00	3.96820E-11	0.00000E+00
98	0.00000E+00	3.96820E-11	0.00000E+00
99	3.84488E-01	8.08513E-01	4.23545E-06
101	2.40927E-06	2.40891E-06	2.28791E-06
102	9.26345E-06	2.95266E-06	9.37161E-06
103	5.07649E-06	4.80719E-07	5.59066E-06
104	3.26251E-06	2.88762E-06	3.99122E-06
105	0.00000E+00	0.00000E+00	0.00000E+00
106	5.22614E-06	4.78864E-06	4.78342E-06
111	8.81476E-06	4.76928E-07	8.38667E-06
114	4.40000E-01	4.40000E-01	4.40000E-01
116	6.82202E-06	0.00000E+00	2.52912E-08
118	0.00000E+00	3.39784E-06	7.28004E-06
119	7.78198E-06	7.45953E-06	5.09764E-06

Table 2 – (continuation).

# of function evaluations	AL-CSearch	AL-BOBYQA	AL-NMead
10^1 to 10^2	6.4 %	4.3 %	0.0 %
10^2 to 10^3	17.0 %	34.0 %	21.3 %
10^3 to 10^4	27.7 %	29.8 %	34.0 %
10^4 to 10^5	19.1 %	14.9 %	14.9 %
10^5 to 10^6	12.8 %	8.5 %	10.6 %
10^6 to 10^7	0.0 %	0.0 %	12.8 %
failure 1	10.6 %	0.0 %	0.0 %
failure 2	0.0 %	2.1 %	0.0 %
failure 3	6.4 %	6.4 %	6.4 %

Table 3 – Function evaluations for 47 problems of Hock-Schittkowski.

classification	AL-CSearch	AL-BOBYQA	AL-NMead
1st	12.8%	57.5%	25.5%
2nd	38.3%	31.9%	21.3%
3rd	31.9%	2.1%	46.8%
failures	17.0%	8.5%	6.4%

Table 4 – Comparison of the subsolvers' performances for 47 problems of Hock-Schittkowski.

a good precision for the simulated area with a small amount of computational effort. We could put the figure of interest inside a very large rectangle, and keep this rectangle fixed within iterations. But, in this case, the computational effort would be possibly bigger.

We considered unions and intersections between three types of figures:

- rectangles, defined by 4 parameters: the coordinates (x, y) of the bottom left vertex, the height and the width;
- circles, defined by 3 parameters: the coordinates of the center and the radius;
- ellipses, defined by 6 parameters, a, b, c, d, e and f, by the formulae $ax^2 + 2bxy + cy^2 + dx + ey + f \leq 0$.

Suppose that we want the point y to be inside some figure A (or B). This is achieved imposing a constraint of type $c_A(y) \leq 0$ (or $c_B(y) \leq 0$). If we want y to lie inside the intersection between figures A and B, we ask that $c_A(y) \leq 0$ and $c_B(y) \leq 0$, so that each point defines two inequality constraints, $2n_p$ at all. On the other hand, the number of constraints defined by unions are n_p, since each point must be inside figure A or figure B. This is achieved imposing that $c(x) = \min\{c_A(x), c_B(x)\} \leq 0$.

We firstly considered the problem of minimizing the

1. intersection of two circles,
2. intersection between a rectangle and a ellipse,
3. union of a rectangle and a ellipse and
4. union of two ellipses.

In the four cases the problem consists of finding the smallest intersection (cases 1 and 2) or union (cases 3 and 4) area that contains a given set of points. In all the problems we considered the same set of $n_p = 10$ points given in Table 5.

We adopted $\varepsilon_{feas} = \delta_{opt} = 10^{-4}$. The initial guesses for circles and ellipses were chosen as circles of radius one centered at the origin and for rectangles, squares of side 2 centered at the origin. The value of n_p in our tests is 10, and the points are

i	(x_i, y_i)		i	(x_i, y_i)
1	(-2.0,-1.0)		6	(0.5,1.5)
2	(-1.5,1.0)		7	(1.0,0.5)
3	(-1.0,1.5)		8	(1.5,1.0)
4	(-1.0,-1.0)		9	(1.0,-2.0)
5	(-1.0,-2.0)		10	(2.0,-1.0)

Table 5 – Points that have to be inside the figure whose simulated area is being minimized.

The density of points generated in the simulation is 10^5 per unit of area, but the maximum of points allowed is 10^7. This means that, if the rectangle that contains the desired figure (used in the simulation of the area) has dimensions $a \times b$, then the number of generated points is $\min\{10^5 ab, 10^7\}$. The initial seeds used in the random points generator are the same at each call of the simulator, so the simulated area is always the same for the same parameters.

It is simple to find the smallest rectangle that contains a specific rectangle, circle or ellipse. The smallest rectangle that contains a given rectangle is the proper rectangle; for a circle or an ellipse it is the one that is tangent to these figures on four points that can be explicitly found. On the other hand, the area A of a figure F can be simulated generating n_g random points inside a rectangle with area A_R that contains F and counting the amount of points that lie inside the figure, n_i, so we have that $A \approx A_R n_i / n_g$. We use these two ideas to compute our objective function according to the following rules:

- In the case of intersection between figures F and G, where F and G can be a rectangle, a circle or an ellipse, we compute the smallest rectangles R_F and R_G that contains each figure, and use the rectangle with smallest area between them to simulate the area of the intersection, generating

random points inside it and counting the amount of points that lie on F and G. We are using the fact that the intersection of figures F and G is inside R_F and is also inside R_G.

- In the case of union between two rectangles, a rectangle and a circle or two circles, we add the areas of the figures, since the expression to these areas is well known, and subtract the area of the intersection, which is computed by the method above. We are using the fact that the area of the union is the sum of the area of the figures minus the area of the intersection.

- In the case of union between a circle and an ellipse, we simulate the area of the figure formed by the ellipse minus the circle, making the simulation inside the smallest rectangle that contains the ellipse. We count how many of the random points lie inside the ellipse but outside the circle, so the desired area is approximately the area of the circle plus the simulated area. The same method is used to simulate the area between a rectangle and an ellipse.

- In the case of union between two ellipses, we find the smallest rectangle that contains both ellipses, and simulate the area of the union counting the amount of random points that lie inside the first or the second (or both) ellipse.

Of course, there are many methods to simulate the considered areas, and we are not claiming that the one we chose is the best. More efficient techniques to simulate these areas is a topic under consideration, and can be the scope of future works.

Figures 1–12 show the results obtained in some instances of area problems. The A value is the area of the figure, while $feval$ is the number of function evaluations performed to find the solution.

Table 6 shows the objective function (the area) found and the number of function evaluations performed for each of the subalgorithms in the attempt of minimizing the area of different types of figures containing the 10 points of Table 5. I and U stand for intersection and union, while R, C and E mean rect-

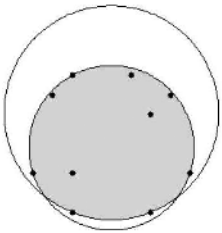

Figure 1 – Intersection of two circles, Algorithm: AL-CSearch, $A = 13.247$, $feval = 4277$.

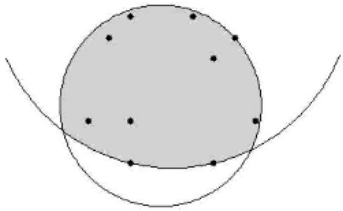

Figure 2 – Intersection of two circles, Algorithm: AL-BOBYQA, $A = 15.004$, $feval = 6283$.

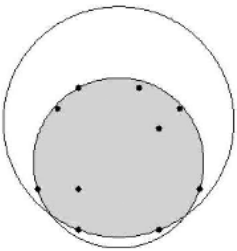

Figure 3 – Intersection of two circles, Algorithm: AL-NMead, $A = 13.247$, $feval = 6099$.

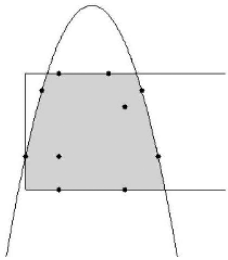

Figure 4 – Intersection between a rectangle and an ellipse, Algorithm: AL-CSearch, $A = 12.574$, $feval = 6260$.

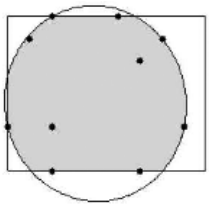

Figure 5 – Intersection between a rectangle and an ellipse, Algorithm: AL-BOBYQA, $A = 12.564$, $feval = 4177$.

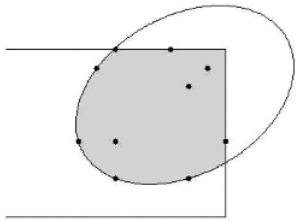

Figure 6 – Intersection between a rectangle and an ellipse, Algorithm: AL-NMead, $A = 13.174$, $feval = 11379$.

angle, circle and ellipse, respectively. The last line shows the average area and average number of function calls among all problems.

The simulated area is (slightly) discontinuous with respect to the parameters that define the figures. The reason is that this area can take only a finite number of values, depending on the number of points that belong to the region.

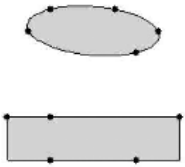

Figure 7 – Union of rectangle and ellipse, Algorithm: AL-CSearch, $A = 6.753$, $feval = 9955$.

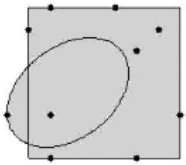

Figure 8 – Union of rectangle and ellipse, Algorithm: AL-BOBYQA, $A = 12.887$, $feval = 6059$.

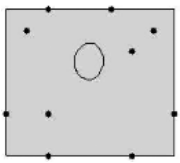

Figure 9 – Union of rectangle and ellipse, Algorithm: AL-NMead, $A = 14.000$, $feval = 7251$.

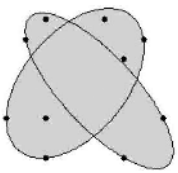

Figure 10 – Union of two ellipses, Algorithm: AL-CSearch, $A = 12.237$, $feval = 16154$.

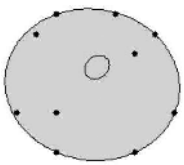

Figure 11 – Union of two ellipses, Algorithm: AL-BOBYQA, $A = 13.339$, $feval = 8191$.

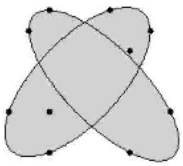

Figure 12 – Union of two ellipses, Algorithm: AL-NMead, $A = 11.363$, $feval = 18881$.

The impact of the discontinuities decreases with the number of points used in the simulation. (Note that the computer time of function evaluations is proportional to that number). Besides, the function $c(x)$ defined for the union between figures has discontinuous derivatives. By these reasons, these problems do not satisfy the smoothness requirements of the convergence theorems presented in this paper. Despite of this fact, in all the cases our algorithm was able to find what seems to be a local minimizer. Since there are several local minimizers, the results show us how each subalgorithm behaved with respect to finding the smallest possible area and computer work (function evaluations) required for achieving a solution. The figures with the smallest and the largest simulated

	area			*feval*		
figure	AL-CSearch	AL-BOBYQA	AL-NMead	AL-CSearch	AL-BOBYQA	AL-NMead
I RR	14.001	13.999	14.003	2709	1535	4959
I RC	12.722	13.467	12.572	6035	3629	7784
I CC	13.247	15.004	13.247	4277	6283	6099
I RE	12.574	12.564	13.174	6260	4177	11379
I CE	13.211	12.973	13.207	5020	7201	11132
I EE	12.058	13.519	13.115	6881	13156	12710
U RR	11.500	12.250	11.499	4312	1745	5717
U RC	13.247	13.894	13.231	2788	874	5466
U CC	13.082	13.890	13.081	2636	4021	4798
U RE	6.723	12.887	14.000	9955	6059	7251
U CE	12.459	13.894	13.333	6715	2042	11688
U EE	12.237	13.399	11.363	16154	8191	18881
average	12.255	13.478	12.985	6145	4909	8989

Table 6 – Results for the minimum area problems.

area were found by the Augmented Lagrangian algorithm with, respectively, Coordinate Search and BOBYQA plus Coordinate Search for the subproblems, and are shown in Figures 7 and 2. In terms of the average area among all figures, shown in the last line of Table 6, AL-CSearch had the best performance, followed by AL-NMead and AL-BOBYQA. On the other hand, AL-BOBYQA required less function evaluations on average, while AL-NMead was the most expensive option.

6 Final remarks

In this paper we defined new derivative-free algorithms for minimizing finite-dimensional functions with two levels of constraints, following the Augmented Lagrangian approach of [1], in such a way that theoretical convergence results are preserved. We introduced an algorithm for the case in which lower-level constraints are defined by bounds on the variables, allowing us to compare the relative efficiency of different box-constraints derivative-free solvers in the context of Augmented Lagrangian subproblems. Then, we defined a new algorithm for dealing with arbitrary lower-level constraints. This algorithm generates subproblems that may be solved by procedures exhibiting an admissibility property. Admissible algorithms for the subproblems may include many of the

recently introduced methods for constrained optimization when the constraints, generally speaking, admit extreme penalty approaches. Here, we showed that, besides the existing admissible algorithms for solving subproblems, the Augmented Lagrangian algorithm with box constraints considered first in this paper (Algorithm 2) may also be considered an admissible algorithm. The practical consequences of this observation remain to be exploited in future research.

From the computational point of view, we introduced a family of Volume Optimization problems, in which we optimize Monte Carlo approximations of volumes subject to constraints. We solved several toy problems of this class, which allowed us to compare different alternatives for solving box-constraint minimization problems in the context of Augmented Lagrangian subproblems without derivatives. We believe that these problems emulate some of the common characteristics of real-life derivative-free optimization problems, especially those originated in simulations.

Acknowledgements. We are indebted to the anonymous referees whose comments helped us to improve the first version of this paper.

REFERENCES

[1] R. Andreani, E.G. Birgin, J. M. Martínez and M. L. Schuverdt, *On Augmented Lagrangian Methods with general lower-level constraints.* SIAM Journal on Optimization, **18** (2007), 1286–1309.

[2] R. Andreani, J.M. Martínez and M.L. Schuverdt, *On the relation between the Constant Positive Linear Dependence condition and quasinormality constraint qualification.* Journal of Optimization Theory and Applications, **125** (2005), 473–485.

[3] C. Audet and J.E. Dennis Jr, *Mesh adaptive direct search algorithms for constrained optimization.* SIAM Journal on Optimization, **17** (2006), 188–217.

[4] C. Audet, J.E. Dennis Jr. and S. Le Digabel, *Globalization strategies for Mesh Adaptive Direct Search*. Computational Optimization and Applications, **46** (2010), 193–215.

[5] E.G. Birgin, C. Floudas and J.M. Martínez, *Global minimization using an Augmented Lagrangian method with variable lower-level constraints*. Mathematical Programming, **125** (2010), 139–162.

[6] R.E. Caflisch, *Monte Carlo and quasi-Monte Carlo methods*, Acta Numerica, Cambridge University Press, **7** (1998), 1–49.

[7] A.R. Conn, N.I.M. Gould and Ph.L. Toint, *A globally convergent Augmented Lagrangian algorithm for optimization with general constraints and simple bounds.* SIAM Journal on Numerical Analysis, **28** (1991), 545–572.

[8] A.R. Conn, N.I.M. Gould and Ph.L. Toint.,*Trust Region Methods*. MPS/ SIAM Series on Optimization, SIAM, Philadelphia (2000).

[9] A.R. Conn, K. Scheinberg and L. N. Vicente, *Introduction to Derivative-Free Optimization*. MPS-SIAM Series on Optimization, SIAM, Philadelphia (2009).

[10] A.L. Custódio and L.N. Vicente, *Using sampling and simplex derivatives in pattern search methods.* SIAM Journal on Optimization, **18** (2007), 537–555.

[11] B.P. Demidovich and I.A. Maron, *Computational Mathematics.* Mir Publishers, Moscow, (1987).

[12] M.R. Hestenes, *Multiplier and gradient methods.* Journal of Optimization Theory and Applications, **4** (1969), 303–320.

[13] W. Hock and K. Schittkowski, *Test Examples for Nonlinear Programming Codes.* Lecture Notes in Economics and Mathematical Systems, Springer, **187** (1981).

[14] T.G. Kolda, R.M. Lewis and V. Torczon, A generating set direct search augmented Lagrangian algorithm for optimization with a combination of general and linear constraints. Technical Report, SAND2006–5315, Sandia National Laboratories, (2006).

[15] T.G. Kolda, R.M. Lewis and V. Torczon, *Stationarity results for generating set search for linearly constrained optimization*. SIAM Journal on Optimization, **17** (2006), 943–968.

[16] R.M. Lewis and V. Torczon, *Pattern search algorithms for bound constrained minimization.* SIAM Journal on Optimization, **9** (1999), 1082–1099.

[17] R.M. Lewis and V. Torczon, *Pattern search algorithms for linearly constrained minimization.* SIAM Journal on Optimization, **10** (2000), 917–941.

[18] R.M. Lewis and V. Torczon, *A globally convergent Augmented Lagrangian pattern search algorithm for optimization with general constraints and simple bounds.* SIAM Journal on Optimization, **12** (2002), 1075–1089.

[19] R.M. Lewis and V. Torczon, A direct search approach to nonlinear programming problems using an Augmented Lagrangian method with explicit treatment of linear constraints. Technical Report WM-CS-2010-01, College of William & Mary, Department of Computer Sciences, (2010).

[20] S. Lucidi, M. Sciandrone and P. Tseng, *Objective derivative-free methods for constrained optimization*. Mathematical Programming, **92** (2002), 37–59.

[21] O.L. Mangasarian and S. Fromovitz, *The Fritz-John necessary optimality conditions in presence of equality and inequality constraints.* Journal of Mathematical Analysis and Applications, **17** (1967), 37–47.

[22] J.A. Nelder and R. Mead, *A simplex method for function minimization.* The Computer Journal, **7** (1965), 308–313.

[23] L.G. Pedroso, *Programação não linear sem derivadas.* PhD thesis, State University of Campinas, Brazil, (2009).

[24] M.J.D. Powell, *A method for nonlinear constraints in minimization problems,* in Optimization, R. Fletcher (ed.). Academic Press, New York, NY, pp. 283–298, (1969).

[25] M.J.D. Powell, *The BOBYQA algorithm for bound constrained optimization without derivatives.* Cambridge NA Report NA2009/06, University of Cambridge, Cambridge, (2009).

[26] L. Qi and Z. Wei, *On the constant positive linear dependence condition and its application to SQP methods.* SIAM Journal on Optimization, **10** (2000), 963–981.

[27] R.T. Rockafellar, *A dual approach for solving nonlinear programming problems by unconstrained optimization*. Mathematical Programming, **5** (1973), 354–373.

[28] R.T. Rockafellar, *Lagrange multipliers and optimality.* SIAM Review, **35** (1993), 183–238.