Technical Report WM-CS-2010-01

College of
William & Mary
Department of Computer Science

WM-CS-2010-01

A Direct Search Approach to Nonlinear Programming Problems Using an Augmented Lagrangian Method with Explicit Treatment of Linear Constraints

Robert Michael Lewis and Virginia Torczon

January 2010

Submitted to SIAM Journal on Optimization

A DIRECT SEARCH APPROACH TO NONLINEAR PROGRAMMING PROBLEMS USING AN AUGMENTED LAGRANGIAN METHOD WITH EXPLICIT TREATMENT OF LINEAR CONSTRAINTS

ROBERT MICHAEL LEWIS* AND VIRGINIA TORCZON†

Abstract. We consider solving nonlinear programming problems using an augmented Lagrangian method that makes use of derivative-free generating set search to solve the subproblems. Our approach is based on the augmented Lagrangian framework of Andreani, Birgin, Martínez, and Schuverdt which allows one to partition the set of constraints so that one subset can be left explicit, and thus treated directly when solving the subproblems, while the remaining constraints are incorporated in the augmented Lagrangian. Our goal in this paper is to show that using a generating set search method for solving problems with linear constraints, we can solve the linearly constrained subproblems with sufficient accuracy to satisfy the analytic requirements of the general framework, even though we do not have explicit recourse to the gradient of the augmented Lagrangian function. Thus we inherit the analytical features of the original approach (global convergence and bounded penalty parameters) while making use of our ability to solve linearly constrained problems effectively using generating set search methods. We need no assumption of nondegeneracy for the linear constraints. Furthermore, our preliminary numerical results demonstrate the benefits of treating the linear constraints directly, rather than folding them into the augmented Lagrangian.

Key words. Nonlinear programming, augmented Lagrangian methods, constraint qualifications, constant positive linear dependence, linear constraints, direct search, generating set search, generalized pattern search, derivative-free methods.

AMS subject classifications. 90C56, 90C30, 65K05

1. Introduction.

We consider solving nonlinear programming problems (NLPs) using an augmented Lagrangian approach in which the subproblems are solved inexactly via generating set search without the explicit use of derivatives. The optimization problem of interest is

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & h(x) = 0, \\ & g(x) \le 0, \\ & a_i^T x = b_i \quad \text{for } i \in \mathcal{E}, \\ & a_i^T x \le b_i \quad \text{for } i \in \mathcal{I}, \end{array} \tag{1.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$, $h : \mathbb{R}^n \to \mathbb{R}^m$, $g : \mathbb{R}^n \to \mathbb{R}^p$, $\mathcal{E}$ and $\mathcal{I}$ are finite index sets for the linear equality and inequality constraints, respectively, and $a_i \in \mathbb{R}^n$ and $b_i \in \mathbb{R}^n$ for all $i \in \mathcal{E}$ and $i \in \mathcal{I}$. We assume that all these functions possess continuous first derivatives on a sufficiently large open domain. However, we also assume that neither explicit nor sufficiently accurate estimates of the derivatives of f, h, and g are available.

To solve problem (1.1), we make use of the augmented Lagrangian framework of Andreani, Birgin, Martínez, and Schuverdt presented in [1]. This approach allows the partial elimination of constraints by incorporating them in the augmented Lagrangian

*Department of Mathematics, College of William & Mary, P.O. Box 8795, Williamsburg, Virginia, 23187-8795; rmlewi@wm.edu. This work was supported by the National Science Foundation under Grant No. DMS-0713812.

†Department of Computer Science, College of William & Mary, P.O. Box 8795, Williamsburg, Virginia, 23187-8795; va@cs.wm.edu.

while retaining the remaining constraints and treating them explicitly. In the case we consider here, the linear constraints in (1.1) are treated explicitly.

The general structure of augmented Lagrangian methods may be viewed as a doubly nested loop. The outer loop consists of successive approximate minimizations of an augmented Lagrangian, where the multiplier estimates and penalty parameter may change from iteration to iteration. The inner loop comprises an iterative minimization of the augmented Lagrangian to a prescribed degree of accuracy. The framework in [1] allows one to leave a subset of the constraints out of the augmented Lagrangian and treat this subset directly in the inner iteration. The idea is that if one has effective computational methods for handling these so-called lower-level constraints, it is better to deal with such constraints directly rather than folding them into the augmented Lagrangian [2, Section 2.4].

Our motivation for partitioning the constraints into two sets, the upper-level nonlinear constraints and the lower-level linear constraints (including bounds), is that we have an effective implementation [14, 17] of a generating set search (GSS) method for solving problems with linear constraints [15, 13]. We prefer to leave linear constraints explicit when solving the subproblems since we can take advantage of their geometric structure in the search for a Karush–Kuhn–Tucker (KKT) point. Moreover, we can do so without any assumption of linear independence of the linear constraints active at the solutions of the subproblems. This direct treatment of the linear constraints has the obvious further advantage of reducing the number of Lagrange multipliers that must be explicitly estimated. One goal of this paper is to demonstrate that there can be a computational advantage to this strategy in the presence of general linear constraints, particularly when the linear constraints exhibit degeneracy.

This line of algorithmic development continues our work in [16, 12] on combining augmented Lagrangian approaches and direct search methods. The work presented here makes use of the desirable analytical properties of the framework outlined in [1]; namely, that feasible limit points which satisfy a fairly mild constraint qualification must be KKT points and that under suitable local conditions, the penalty parameter remains bounded. Here we show that generating set search can solve the linearly constrained subproblems with sufficient accuracy to satisfy the demands of the analysis in [1]—even without explicit recourse to derivatives for f, h, and g.

Section 2 describes our previous approaches to solving nonlinear programming problems using augmented Lagrangian methods and explains the motivation for the new approach presented here. Section 3 reviews the augmented Lagrangian algorithm framework of [1]. Section 4 presents the salient features of GSS methods for solving linearly constrained problems. In Section 5 we show that we can minimize the augmented Lagrangian subproblems with sufficient accuracy using GSS methods. Section 6 reviews the results in [1] that apply to our approach. Section 7 contains some preliminary numerical results which illustrate that our approach is not only analytically sound, but also can be competitive in practice. These results confirm a long-standing (and eminently sensible) conjecture that direct treatment of the linear constraints can improve the performance of an augmented Lagrangian approach to solving NLPs.

Notation. Given vectors u and v, let $\max(u, v)$ denote the componentwise maximum of u and v (i.e., the i-th component of $\max(u, v)$ is $\max\{u_i, v_i\}$). Unless explicitly indicated otherwise, $\|\cdot\|$ denotes the standard Euclidean norm. If $K = \{k_j\} \subset \mathbb{N}$ satisfies $k_{j+1} > k_j$ for all $j = 0, 1, 2, \ldots$, then we define $\lim_{k \in K} x^{(k)} = \lim_{j \to \infty} x^{(k_j)}$. To distinguish between quantities associated with the outer iterations and those associated with the inner iterations required to approximately solve subproblem (2.2), we use the counter k for the outer iterations and the counter j for the inner iterations. Quantities associated with the outer iterations are then denoted $x^{(k)}$ while quantities associated with the inner iterations are denoted x_j.

2. Background. We have previously studied augmented Lagrangian approaches to solving nonlinear programs in the absence of reliable derivative information in [16, 12]. Throughout we have sought to deal explicitly with those constraints for which we have efficient solution techniques. But the work here differs in several respects. We begin with the following augmented Lagrangian L corresponding to the constraints $h(x) = 0$ and $g(x) \leq 0$: for $x \in \mathbb{R}^n$, $\rho > 0$, $\lambda \in \mathbb{R}^m$, $\mu \in \mathbb{R}^p$, $\mu \geq 0$,

$$L(x; \lambda, \mu, \rho) = f(x) + \frac{\rho}{2}\left(\| \, h(x) + \lambda/\rho \, \|^2 + \| \max(0, \; g(x) + \mu/\rho) \, \|^2 \right). \tag{2.1}$$

The lower-level linear constraints in (1.1) are absent from the augmented Lagrangian since we treat these constraints explicitly.

Our first approach to partitioning the set of constraints into upper- and lower-level constraints, described in [16], treated the simpler case where only bound constraints are dealt with explicitly. All remaining constraints, including general linear constraints, were folded into the augmented Lagrangian. Generating set search was then applied to the resulting bound constrained subproblems inside the augmented Lagrangian framework of [7].

In [12] we combined generating set search with the augmented Lagrangian approach in [6]. The linearly constrained subproblems

$$\begin{array}{lll} \text{minimize} & L(x; \lambda, \mu, \rho) & \\ \text{subject to} & a_i^T x = b_i & i \in \mathcal{E}, \\ & a_i^T x \leq b_i & i \in \mathcal{I}, \end{array} \tag{2.2}$$

were then solved using generating set search methods for linearly constrained optimization. The direct search augmented Lagrangian method in the Matlab GADS Toolbox [21] is based on this approach.

The augmented Lagrangian frameworks of [7, 6], as well as that from [1] of interest here, require only approximate minimization of the augmented Lagrangian in the subproblems. The accuracy of the minimization in the subproblems is measured in terms of stationarity conditions that involve the first derivative of the Lagrangian.

The key to integrating generating set search with such augmented Lagrangian approaches lies in showing that the subproblems can be solved to sufficient accuracy, even though we do *not* have explicit knowledge of the gradient of the Lagrangian. To do so, we use the fact than when applying generating set search to bound and linearly constrained problems, at an identifiable subsequence of iterates, suitable measures of stationarity are $O(\Delta_j)$, where Δ_j is an algorithmic parameter used to determine the lengths of the steps in all GSS algorithms. Thus by monitoring the value of Δ_j, we can ensure that the subproblems are solved to sufficient accuracy, even though we have no explicit knowledge of the gradient of the Lagrangian.

In the present work we make use of the stationarity results from [17] and the more general augmented Lagrangian framework proposed in [1]. From our perspective the latter enjoys two clear advantages. The first advantage is that [1] establishes first-order stationarity using a weaker constraint qualification than that used in [7, 6]. Specifically, the result [1, Theorem 4.2], which appears here as Theorem 6.2, makes use of the constant positive linear dependence (CPLD) constraint qualification [20] rather than the linear independence constraint qualification (LICQ) used in [7, 6]. The second advantage of the approach in [1] is that boundedness of the multiplier estimates is enforced, as discussed in [1, Sections 1–2]. This second feature has salutary theoretical and computational consequences, which we review in Section 3.2.

3. Description of the augmented Lagrangian framework.

We next review the augmented Lagrangian framework presented in [1, Section 3], found here as Algorithm 3.1.

Step 0. Set up. Let $x^{(0)} \in \mathbb{R}^n$ be an arbitrary initial point. Let $\tau \in [0, 1)$, $\gamma > 1$, $\rho^{(1)} > 0$, $-\infty < \bar{\lambda}_{\min} \leq \bar{\lambda}^{(1)} \leq \bar{\lambda}_{\max} < \infty$, $0 \leq \bar{\mu}^{(1)} \leq \bar{\mu}_{\max} < \infty$. Finally, let $\{\varepsilon^{(k)}\} \subset \mathbb{R}$, $\varepsilon^{(k)} \geq 0$, be a sequence of tolerance parameters such that $\lim_{k\to\infty} \varepsilon^{(k)} = 0$.

Step 1. Initialization. Set $k = 1$ and $\sigma^{(0)} = \max\left(0, g(x^{(0)})\right)$.

Step 2. Solving the subproblem. Find (if possible) an $x^{(k)} \in \mathbb{R}^n$ for the problem (2.2) such that there exist $v^{(k)}$ and $u^{(k)}$ satisfying (3.1)–(3.4). If it is not possible to find an $x^{(k)}$ satisfying (3.1)–(3.4), terminate.

Step 3. Update the multiplier estimates.

- Set $\lambda^{(k+1)} = \bar{\lambda}^{(k)} + \rho^{(k)} h(x^{(k)})$.
- From $\lambda^{(k+1)}$ compute $\bar{\lambda}^{(k+1)}$ satisfying $\bar{\lambda}_{\min} \leq \bar{\lambda}^{(k+1)} \leq \bar{\lambda}_{\max}$, e.g., by projecting $\lambda^{(k+1)}$ onto $\{\, \bar{\lambda} \mid \bar{\lambda}_{\min} \leq \bar{\lambda} \leq \bar{\lambda}_{\max} \,\}$.
- Set

$$\sigma^{(k)} = \max\left(g(x^{(k)}),\ -\bar{\mu}^{(k)}/\rho^{(k)}\right),$$
$$\mu^{(k+1)} = \max\left(0,\ \bar{\mu}^{(k)} + \rho^{(k)} g(x^{(k)})\right).$$

- From $\mu^{(k+1)}$ compute $\bar{\mu}^{(k+1)}$ satisfying $0 \leq \bar{\mu}^{(k+1)} \leq \bar{\mu}_{\max}$, e.g., by projecting $\mu^{(k+1)}$ onto $\{\, \bar{\mu} \mid 0 \leq \bar{\mu} \leq \bar{\mu}_{\max} \,\}$.

Step 4. Update the penalty parameter. If

$$\max\left\{\left\| h(x^{(k)}) \right\|_\infty, \left\| \sigma^{(k)} \right\|_\infty\right\} \leq \tau \max\left\{\left\| h(x^{(k-1)}) \right\|_\infty, \left\| \sigma^{(k-1)} \right\|_\infty\right\}$$

then $\rho^{(k+1)} = \rho^{(k)}$; else, $\rho^{(k+1)} = \gamma\rho^{(k)}$.

Step 5. Begin a new outer iteration. Set $k = k + 1$. Go to Step 2.

Algorithm 3.1
The augmented Lagrangian framework.

Given the definition in (2.1) of the augmented Lagrangian with respect to the upper-level constraints, at each outer iteration of the augmented Lagrangian algorithm, the task of the inner iteration is that of solving inexactly, but to a suitable degree, the subproblem (2.2) using bounded Lagrangian multiplier estimates $\bar{\lambda}$ and $\bar{\mu}$ along with a penalty parameter ρ. Next we discuss the conditions on the inexact solution of (2.2) and then review how boundedness of $\bar{\lambda}$ and $\bar{\mu}$ is enforced.

3.1. Conditions on the inexact solution of the subproblem. At each outer iteration k, the inner iteration is required to solve (2.2) inexactly. Specifically, for the analysis in [1] to hold, the inner iteration must find an $x^{(k)} \in \mathbb{R}^n$ such that there exist $v^{(k)} \in \mathbb{R}^{|\mathcal{E}|}$ and $u^{(k)} \in \mathbb{R}^{|\mathcal{I}|}$ satisfying

$$\left\| \nabla L(x^{(k)}; \bar{\lambda}^{(k)}, \bar{\mu}^{(k)}, \rho^{(k)}) + \sum_{i \in \mathcal{E}} v_i^{(k)} a_i + \sum_{i \in \mathcal{I}} u_i^{(k)} a_i \right\| \le \varepsilon_1^{(k)}, \tag{3.1}$$

where

$$u_i^{(k)} \ge 0 \quad \text{and} \quad a_i^T x^{(k)} - b_i \le \varepsilon_2^{(k)} \qquad \forall\, i \in \mathcal{I}, \tag{3.2}$$

$$a_i^T x^{(k)} - b_i < -\varepsilon_2^{(k)} \quad \Rightarrow \quad u_i^{(k)} = 0 \qquad \forall\, i \in \mathcal{I}, \tag{3.3}$$

and

$$\left(\sum_{i \in \mathcal{E}} (a_i^T x^{(k)} - b_i)^2 \right)^{1/2} \le \varepsilon_3^{(k)}, \tag{3.4}$$

where $0 \le \varepsilon_1^{(k)}, \varepsilon_2^{(k)}, \varepsilon_3^{(k)} \le \varepsilon^{(k)}$. These conditions are relaxations of the KKT conditions for the problem of minimizing the augmented Lagrangian L subject to the condition $x \in \{ x \in \mathbb{R}^n \mid a_i^T x = b_i, i \in \mathcal{E}, \text{ and } a_i^T x \le b_i, i \in \mathcal{I} \}$.

3.2. Bounding the Lagrange multiplier estimates. As discussed in [1, Section 3], the bounded Lagrange multiplier estimates, denoted by $\bar{\lambda}$ and $\bar{\mu}$, are required to satisfy

$$\bar{\lambda}_{\min} \le \bar{\lambda} \le \bar{\lambda}_{\max} < \infty \quad \text{and} \quad 0 \le \bar{\mu} \le \bar{\mu}_{\max} < \infty.$$

This is enforced by projecting the updated multiplier estimates onto a compact box at the end of each outer iteration. Among other things, enforcing the boundedness of the multiplier estimates ensures that the augmented Lagrangian approach will, in the worst case, behave like a quadratic penalty function.

3.3. Progress in terms of feasibility and complementarity. We close by recapping the observations in [1, Section 2] on assessing progress towards both *feasibility* and *complementarity* with respect to the upper-level constraints. For the equality constraints $h(x) = 0$, complementarity is not an issue and infeasibility with respect to any given equality constraint h_i is measured simply by $|\, h_i(x) \,|$.

Matters are more complicated for the inequality constraints $g(x) \le 0$ because of the need to address the issue of complementarity; to wit, $\mu_i = 0$ if $g_i(x) < 0$. This leads to the following measure of infeasibility and noncomplementarity with respect to the inequality constraints $g(x)$:

$$\sigma(x, \mu, \rho) = \max(g(x), -\mu/\rho). \tag{3.5}$$

Given the requirements $\mu \ge 0$ and $\rho > 0$ it follows that $\mu/\rho \ge 0$. Then $\sigma_i(x, \mu, \rho) = 0$ when either $g_i(x) = 0$ (i.e., x is feasible with respect to the constraint g_i) or $g_i(x) < 0$ and $\mu_i = 0$ (i.e., strict complementarity holds for the constraint g_i). Thus, σ_i is a measure of the trade-off between feasibility and complementarity with respect to the constraint g_i.

If the inner iteration has not made sufficient progress in terms of both feasibility (with respect to h and g) and complementarity (with respect to g), then the penalty parameter ρ is increased by a fixed factor before proceeding to the next outer iteration.

4. GSS methods for linearly constrained problems. We consider GSS applied to the linearly constrained problem

$$\begin{array}{lll} \text{minimize} & \Phi(x) & \\ \text{subject to} & a_i^T x = b_i, & i \in \mathcal{E}, \\ & a_i^T x \le b_i, & i \in \mathcal{I}. \end{array} \tag{4.1}$$

Let $\Psi = x \in \left\{ x \in \mathbb{R}^n \mid a_i^T x = b_i, i \in \mathcal{E}, \text{ and } a_i^T x \le b_i, i \in \mathcal{I} \right\}$ denote the feasible region for (4.1). Detailed descriptions of GSS methods for solving linearly constrained problems appear in [15, 13, 14, 17]. The presentation here is based on [17, Algorithm 4.1], which is restated as Algorithm 4.1. The introduction of the parameter $\theta_{\min}$, discussed in Section 5, makes Algorithm 4.1 a slight specialization of [17, Algorithm 4.1].

We recall those features needed to confirm that it is possible to produce an inexact solution to (2.2) that satisfies (3.1)–(3.4). GSS methods for linearly constrained problems are feasible iterates methods, so at every iteration they enforce $x \in \Psi$. If an infeasible initial iterate is given, it is replaced by computing a feasible initial iterate x_0 from which the actual search begins. For instance, if x_0 is infeasible it can be replaced by its Euclidean projection onto the feasible polyhedron [14, Section 8.1].

Maintaining feasibility with respect to the linear equality constraints is straightforward [14, Section 4]. The key to ensuring feasibility with respect to the linear inequality constraints lies in constructing a set of search directions that ensures it is possible to take a sufficiently long step from the iterate x, while remaining feasible. To do so, we monitor the constraints that are active near x. Specifically, for $i \in \mathcal{I}$ let $\mathcal{C}_i = \left\{ y \mid a_i^T y = b_i \right\}$, the affine subspace associated with the ith inequality constraint. Given $x \in \Psi$ and $r \ge 0$, let $\mathcal{I}(x, r) = \left\{ i \in \mathcal{I} \mid \operatorname{dist}(x, \mathcal{C}_i) = \left| b_i - a_i^T x \right| / \| a_i \| \le r \right\}$. The vectors a_i for $i \in \mathcal{I}(x, r)$ are outward-pointing normals to the inequality constraints that contribute to determining the boundary of Ψ within distance r of x. We then define the r-normal cone $N(x, r)$ to be the cone

$$\left\{ z \in \mathbb{R}^n \mid z = \sum_{i \in \mathcal{E}} v_i a_i + \sum_{i \in \mathcal{I}(x,r)} u_i a_i, \ v_i, u_i \in \mathbb{R}, \ u_i \ge 0 \right\}. \tag{4.2}$$

The r-tangent cone $T(x, r)$ is defined to be the cone polar to $N(x, r)$. These two cones are illustrated in Figure 4.1.

What is required, at a minimum, for each iteration j of a GSS method is that the set of possible search directions $\mathcal{D}_j$ contains a *core set of search directions*, denoted $\mathcal{G}_j$, which consists of generators for $T(x_j, r_j)$. Here we assume that the directions in $\mathcal{G}_j$ have been normalized; this is not necessary, as discussed in [13], but it simplifies both our discussion and the analysis without any apparent detriment to performance (see the numerical results in [17, Section 6]). We then use the step-length control parameter $\Delta_j \ge 0$ to control the length of the steps defined by the directions in $\mathcal{D}_j$. Further, if we require $r_j = \Delta_j$, then we ensure that at least for the directions $d \in \mathcal{G}_j$ a step of length Δ_j is feasible [13, Proposition 2.2]; this can be seen in Figure 4.1(a). When the search is unable to find sufficient decrease at any of the trial

Step 0. Initialization. Let $x_0 \in \Psi$ be the initial iterate. Let $\Delta_{\text{tol}} > 0$ be the tolerance used to test for convergence. Let $\Delta_0 > \Delta_{\text{tol}}$ be the initial value of the step-length control parameter. Let $\Delta_0 \leq \Delta_{\max} < \infty$ be an upper bound on the step-length control parameter. Let $r_{\max} > \Delta_{\text{tol}}$ be the maximum distance used to identify nearby constraints ($r_{\max} = +\infty$ is permissible). Let $\alpha > 0$. Let $\varphi_j(\Delta_j) = \alpha \max\left\{ \left| \Phi_{\text{typ}} \right|, \left| \Phi(x_j) \right| \right\} \Delta_j^2$, where $\Phi_{\text{typ}} \neq 0$ is some value that reflects the typical magnitude of the objective for $x \in \Psi$. Let $1 \leq \phi_{\max} < \infty$. Let $0 < \theta_{\min} \leq \theta_{\max} < 1$. Set $\mathcal{S} = \mathcal{U} = \mathcal{U}_T = \emptyset$.

Step 1. Choose search directions. Let $r_j = \min\{r_{\max}, \Delta_j\}$. Choose a set of search directions $\mathcal{D}_j = \mathcal{G}_j \cup \mathcal{H}_j$ satisfying Condition 4.1. Normalize the core search directions in $\mathcal{G}_j$ so that $\| d \| = 1$ for all $d \in \mathcal{G}_j$.

Step 2. Look for decrease. Consider trial steps of the form $x_j + \tilde{\Delta}(d)d$ for $d \in \mathcal{D}_j$, where $\tilde{\Delta}(d)$ is as defined in (4.3), until either finding a $d_j \in \mathcal{D}_j$ that satisfies (4.4) (a successful iteration) or determining that (4.4) is not satisfied for any $d \in \mathcal{G}_j$ (an unsuccessful iteration).

Step 3. Successful Iteration. If there exists $d_j \in \mathcal{D}_j$ such that (4.4) holds, then:

- Set $x_{j+1} = x_j + \tilde{\Delta}(d_j)d_j$.
- Set $\Delta_{j+1} = \min\{\Delta_{\max}, \phi_j \Delta_j\}$, where $1 \leq \phi_j \leq \phi_{\max}$.
- Set $\mathcal{S} = \mathcal{S} \cup \{j\}$.
- If during Step 2 it was determined that (4.4) is not satisfied for any $d \in \mathcal{G}_j$, set $\mathcal{U}_T = \mathcal{U}_T \cup \{j\}$.

Step 4. Unsuccessful Iteration. Otherwise,

- Set $x_{j+1} = x_j$.
- Set $\Delta_{j+1} = \theta_j \Delta_j$, where $\theta_{\min} \leq \theta_j \leq \theta_{\max} < 1$.
- Set $\mathcal{U} = \mathcal{U} \cup \{j\}$.
- Set $\mathcal{U}_T = \mathcal{U}_T \cup \{j\}$.

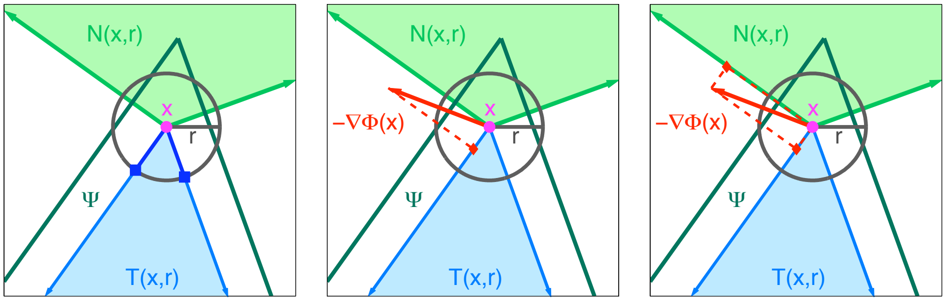
If $\Delta_{j+1} < \Delta_{\text{tol}}$, then terminate.

Step 5. Advance. Increment j by one and go to Step 1.

Algorithm 4.1
A linearly constrained GSS algorithm.

steps defined by $\Delta_j d$, $d \in \mathcal{G}_j$, we keep track of such iterations by adding j to the set $\mathcal{U}_T$ which contains the indices of those iterations for which the search was *tangentially unsuccessful*. Note that it is possible to have an iteration that is both successful and tangentially unsuccessfully if sufficient decrease is found for a $d \in \mathcal{H}_j$ but not for any $d \in \mathcal{G}_j$. If the search is unable to find sufficient decrease at any of the trial steps defined by $d \in \mathcal{D}_j$, then Δ_j is reduced. A traditional strategy is to halve Δ_j.

In order to ensure convergence of linearly constrained GSS, we need to place



(a) The r-normal and r-tangent cones $N(x,r)$ and $T(x,r)$. We use ■ to denote the trial steps defined by $\Delta = r$ and the normalized generators for $T(x,r)$.

(b) Asymptotically, the norm of the projection of $-\nabla\Phi(x)$ onto $T(x,r)$ is less than some constant times $\Delta = r$.

(c) By the polar decomposition, the projection of $-\nabla\Phi(x)$ onto $T(x,r)$ is equal to $-\nabla\Phi(x)$ less its projection onto $N(x,r)$.

FIG. 4.1. *The geometry of linear constraints.*

Condition 4.1 on the sequence of core sets $\{\mathcal{G}_j\}$. Condition 4.1 makes use of the quantity $\kappa(\mathcal{G})$, which appears in [13, (2.1)] and is a generalization of that given in [11, (3.10)]. We make use of the following notation: if $K \subseteq \mathbb{R}^n$ is a convex cone and $z \in \mathbb{R}^n$, then denote the Euclidean norm projection of z onto K by $[z]_K$. Now, for any finite set of vectors $\mathcal{G}$ define

$$\kappa(\mathcal{G}) = \inf_{\substack{z\in\mathbb{R}^n \\ [z]_K \neq 0}} \max_{d\in\mathcal{G}} \frac{z^T d}{\| \, [z]_K \, \| \; \| \, d \, \|}, \quad \text{where } K \text{ is the cone generated by } \mathcal{G}.$$

We place the following condition on the set of search directions $\mathcal{G}_j$.

CONDITION 4.1. *There exists a constant $\kappa_{\min} > 0$, independent of j, such that for every j, $\mathcal{G}_j$ generates $T(x_j, r_j)$, and if $\mathcal{G}_j \neq \{\mathbf{0}\}$, then $\kappa(\mathcal{G}_j) \geq \kappa_{\min}$.*

In the event that $T(x_j, r_j)$ contains a nontrivial lineality space we have freedom in our choice of generators for $T(x_j, r_j)$. The lower bound $\kappa_{\min}$ precludes a subsequence of $\{\mathcal{G}_j\}$ for which $\kappa(\mathcal{G}_j) \to 0$. For further discussion of $\kappa(\cdot)$, see [17, Section 4.1].

Next, we refine the step-length control mechanism to handle those search directions $d \in \mathcal{H}_j = \mathcal{D}_j \setminus \mathcal{G}_j$, since feasibility of $x_j + \Delta_j d$ is ensured only for the directions $d \in \mathcal{G}_j$. Define

$$\tilde{\Delta}(d) = \max \{ \, \Delta \in [0, \Delta_j] \mid x_j + \Delta d \in \Psi \, \}. \tag{4.3}$$

Then for any $d \in \mathcal{D}_j$, the associated trial point is $x_j + \tilde{\Delta}(d)d$. This construction ensures that the full step (with respect to Δ_j) is taken if the resulting trial point is feasible. Otherwise, the trial point is found by taking the longest possible feasible step from x_j along d.

Finally, we define the *sufficient decrease condition* we use. A trial point is considered acceptable only if it satisfies

$$\Phi(x_j + \tilde{\Delta}(d)d) < \Phi(x_j) - \varphi_j(\Delta_j). \tag{4.4}$$

There is considerable latitude in the choice of φ_j; we use $\varphi_j(\Delta) = \alpha_j \Delta^2$ where the sequence $\{\alpha_j\}$ is bounded away from zero for all j. In our current implementation

[17], we use

$$\alpha_j = \alpha \max\left\{|\Phi_{\text{typ}}|, |\Phi(x_j)|\right\}, \tag{4.5}$$

where $\alpha > 0$ is fixed and $\Phi_{\text{typ}} \neq 0$ is some fixed value that reflects the typical magnitude of the objective for feasible inputs. Our default choice is $\Phi_{\text{typ}} = 1$.

We are ready now to state the following results from [17].

THEOREM 4.2. *[17, Theorem 4.2] Suppose that $\nabla\Phi$ is Lipschitz continuous with constant M on Ψ. Consider the iterates produced by the linearly constrained GSS Algorithm 4.1. If $j \in \mathcal{U}_T$ and r_j satisfies $r_j = \Delta_j$, then*

$$\left\| \, [-\nabla\Phi(x_j)]_{T(x_j,r_j)} \, \right\| \leq \frac{1}{\kappa_{\min}}(M + \alpha_j)\Delta_j, \tag{4.6}$$

where $\kappa_{\min}$ is from Condition 4.1 and α_j is from (4.5).

Theorem 4.2 yields a bound on the relative size of the projection of $-\nabla\Phi(x_j)$ onto $T(x_j, r_j)$: if the sequence $\{\Phi(x_j)\}$ is bounded below, then there exists $C > 0$ such that for all $j \in \mathcal{U}_T$ we have

$$\left\| \, [-\nabla\Phi(x_j)]_{T(x_j,r_j)} \, \right\| \leq C\Delta_j,$$

where C depends on $\kappa_{\min}$, M, α, $\left|\, \Phi_{\text{typ}} \right|$, and the upper bound on $\{|\, \Phi(x_j)\,|\}$. The relationship $\left\| \, [-\nabla\Phi(x_j)]_{T(x_j,r_j)} \, \right\| = O(\Delta_j)$ is illustrated in Figure 4.1(b).

The asymptotic behavior of the sequence $\{\Delta_j\}$ is described by Theorem 4.3.

THEOREM 4.3. *[17, Theorem 4.3] Consider the iterates produced by the linearly constrained GSS Algorithm 4.1. Then either $\lim_{j\to\infty} \Delta_j = 0$ or $\lim_{j\to\infty} \Phi(x_j) = -\infty$.*

Theorems 4.2 and 4.3 yield the following first-order stationarity result for (4.1).

THEOREM 4.4. *[17, Theorem 4.4] Let $\nabla\Phi$ be Lipschitz continuous with constant M on Ψ and let $\{x_j\}$ be the sequence of iterates produced by the linearly constrained GSS Algorithm 4.1 with r_j satisfying $r_j = \Delta_j$. If $\{x_j\}$ is bounded, or if Φ is bounded below on Ψ, then*

$$\lim_{j \in \mathcal{U}_T \to \infty} \left\| \, [-\nabla\Phi(x_j)]_{T(x_j,r_j)} \, \right\| = 0.$$

5. The GSS augmented Lagrangian algorithm. In this section we discuss additional considerations that arise when using GSS methods for solving the linearly constrained subproblem (2.2) within the augmented Lagrangian framework of [1]. We first show the connection between the stationarity condition (4.6) and the stationarity conditions (3.1)–(3.4). Afterwords we consider the dependence of the right-hand side of the bound (4.6) on the multiplier estimates and penalty parameter. Algorithm 5.1 is the resulting augmented Lagrangian method.

Proposition 5.1 rewrites the projection of $-\nabla\Phi(x)$ onto $T(x, r)$ in terms of the Lagrangian for (4.1).

PROPOSITION 5.1. *Let x_j be an iterate produced by the linearly constrained GSS Algorithm 4.1 applied to (4.1). Then there exist scalars v_i, $i \in \mathcal{E}$, and u_i, $i \in \mathcal{I}$, such that $u_i \geq 0$ for all $i \in \mathcal{I}$, $u_i = 0$ if $i \notin \mathcal{I}(x_j, r_j)$,*

$$[-\nabla\Phi(x_j)]_{T(x_j,r_j)} = -\nabla\Phi(x_j) - \sum_{i\in\mathcal{E}} v_i a_i - \sum_{i\in\mathcal{I}} u_i a_i,$$

Step 0. Set up. Let $x^{(0)} \in \mathbb{R}^n$ be an arbitrary initial point. Let $\tau \in [0, 1)$, $\gamma > 1$, $\rho^{(1)} > 0$, $-\infty < \bar{\lambda}_{\min} \le \bar{\lambda}^{(1)} \le \bar{\lambda}_{\max} < \infty$, and $0 \le \bar{\mu}^{(1)} \le \bar{\mu}_{\max} < \infty$. Let $0 < \xi_{\min} \le \xi_{\max} < 1$ and $\Delta_{\text{tol}}^{(1)} > 0$. Let $\delta(\bar{\lambda}, \bar{\mu}, \rho)$ be as discussed in connection with the update rule (5.4).

Step 1. Initialization. Set $k = 1$ and $\sigma^{(0)} = \max(0, g(x^{(0)}))$.

Step 2. Solving the subproblem. Apply Algorithm 4.1 to

$$\begin{array}{lll} \text{minimize} & L(x; \bar{\lambda}^{(k)}, \bar{\mu}^{(k)}, \rho^{(k)}) & \\ \text{subject to} & a_i^T x = b_i & i \in \mathcal{E}, \\ & a_i^T x \le b_i & i \in \mathcal{I} \end{array}$$

until the stopping criterion $\Delta_j < \Delta_{\text{tol}}^{(k)}$ is satisfied. Let $x^{(k)}$ be the approximate minimizer so computed; then there exist $v^{(k)}$ and $u^{(k)}$ satisfying (3.1)–(3.4) and (5.1)–(5.3). If it is not possible to find an $x^{(k)}$ satisfying the stopping criterion, terminate.

Step 3. Update the multiplier estimates.

– Set $\lambda^{(k+1)} = \bar{\lambda}^{(k)} + \rho^{(k)} h(x^{(k)})$.

– Compute $\bar{\lambda}^{(k+1)}$ by projecting $\lambda^{(k+1)}$ onto $\{\, \lambda \mid \bar{\lambda}_{\min} \le \lambda \le \bar{\lambda}_{\max} \,\}$.

– Set

$$\begin{aligned} \sigma^{(k)} &= \max\left(g(x^{(k)}),\ -\bar{\mu}^{(k)}/\rho^{(k)} \right), \\ \mu^{(k+1)} &= \max\left(0,\ \bar{\mu}^{(k)} + \rho^{(k)} g(x^{(k)}) \right). \end{aligned}$$

– Compute $\bar{\mu}^{(k+1)}$ by projecting $\mu^{(k+1)}$ onto $\{\, \mu \mid 0 \le \mu \le \bar{\mu}_{\max} \,\}$.

Step 4. Update the penalty parameter. If

$$\max\left\{ \left\| \, h(x^{(k)}) \, \right\|_\infty, \left\| \, \sigma^{(k)} \, \right\|_\infty \right\} \le \tau \max\left\{ \left\| \, h(x^{(k-1)}) \, \right\|_\infty, \left\| \, \sigma^{(k-1)} \, \right\|_\infty \right\}$$

then $\rho^{(k+1)} = \rho^{(k)}$; else, $\rho^{(k+1)} = \gamma \rho^{(k)}$.

Step 5. Update the subproblem stopping tolerance. Choose $\xi^{(k)}$ satisfying $\xi_{\min} \le \xi^{(k)} \le \xi_{\max}$, and set

$$\Delta_{\text{tol}}^{(k+1)} = \xi^{(k)} \Delta_{\text{tol}}^{(k)} / \delta(\bar{\lambda}^{(k+1)}, \bar{\mu}^{(k+1)}, \rho^{(k+1)}).$$

Step 6. Begin a new outer iteration. Set $k = k + 1$. Go to Step 2.

Algorithm 5.1

The augmented Lagrangian framework incorporating GSS.

and

$$\begin{aligned} u_i \ge 0 \quad \textit{and} \quad a_i^T x_j - b_i \le 0 \qquad & \forall\ i \in \mathcal{I}, \\ a_i^T x_j - b_i < -r_j \, \| \, a_i \, \| \quad \Rightarrow \quad u_i = 0 \qquad & \forall\ i \in \mathcal{I}. \end{aligned}$$

Moreover, $a_i^T x_j - b_i = 0$ for all $i \in \mathcal{E}$.

Proof. By the polar decomposition [19] we have

$$[-\nabla\Phi(x_j)]_{T(x_j,r_j)} = -\nabla\Phi(x_j) - [-\nabla\Phi(x_j)]_{N(x_j,r_j)}$$

(as illustrated in Figure 4.1(c)). From the definition of $N(x_j, r_j)$ in (4.2) it follows that there exist v_i (unrestricted in sign) and $u_i \geq 0$ such that

$$[-\nabla\Phi(x_j)]_{N(x_j,r_j)} = \sum_{i\in\mathcal{E}} v_i a_i + \sum_{i\in\mathcal{I}(x_j,r_j)} u_i a_i.$$

Thus, if we set $u_i = 0$ for $i \in \mathcal{I} \setminus \mathcal{I}(x_j, r_j)$, then

$$[-\nabla\Phi(x_j)]_{T(x_j,r_j)} = -\nabla\Phi(x_j) - \sum_{i\in\mathcal{E}} v_i a_i - \sum_{i\in\mathcal{I}} u_i a_i,$$

where $u_i \geq 0$ for all $i \in \mathcal{I}$.

Since Algorithm 4.1 produces only iterates that are feasible with respect to the linear constraints, we are guaranteed to have $a_i^T x - b_i \leq 0$ for all $i \in \mathcal{I}$ and $a_i^T x - b_i = 0$ for all $i \in \mathcal{E}$. In addition, from the definition of $\mathcal{I}(x_j, r_j)$ it follows that

$$i \in \mathcal{I} \setminus \mathcal{I}(x_j, r_j) \Longleftrightarrow a_i^T x_j - b_i < -r_j \,\|\, a_i \,\|\,.$$

□

Proposition 5.2 results from applying Theorem 4.2 to $\Phi(x) = L(x; \bar{\lambda}, \bar{\mu}, \rho)$.

Proposition 5.2. *Consider the sequence of iterates $\{x_j\}$ produced by the linearly constrained GSS Algorithm 4.1 applied to (2.2). Suppose that $M(\bar{\lambda}, \bar{\mu}, \rho)$ is a Lipschitz constant (in x) for $\nabla L(x; \bar{\lambda}, \bar{\mu}, \rho)$ on Ψ. Let $\kappa_{\min}$ be as in Condition 4.1 and α_j as in (4.5). If $j \in \mathcal{U}_T$ and r_j satisfies $r_j = \Delta_j$, then*

$$\|\, [-\nabla L(x_j; \bar{\lambda}, \bar{\mu}, \rho)]_{T(x_j,r_j)} \,\| \leq \frac{1}{\kappa_{\min}} \left(M(\bar{\lambda}, \bar{\mu}, \rho) + \alpha_j \right) \Delta_j.$$

Proposition 5.1 and Proposition 5.2 then yield the following.

Proposition 5.3. *Consider the sequence of iterates $\{x_j\}$ produced by the linearly constrained GSS Algorithm 4.1 applied to (2.2). Suppose that $M(\bar{\lambda}, \bar{\mu}, \rho)$ is a Lipschitz constant (in x) for $\nabla L(x; \bar{\lambda}, \bar{\mu}, \rho)$ on Ψ. Under the hypotheses of Theorem 4.2, if $j \in \mathcal{U}_T$ then there exist scalars v_i, $i \in \mathcal{E}$, and u_i, $i \in \mathcal{I}$, such that $u_i \geq 0$ for all $i \in \mathcal{I}$, $u_i = 0$ if $i \notin \mathcal{I}(x_j, r_j)$,*

$$\left\| \nabla L(x_j; \bar{\lambda}, \bar{\mu}, \rho) + \sum_{i\in\mathcal{E}} v_i a_i + \sum_{i\in\mathcal{I}} u_i a_i \right\| \leq \frac{1}{\kappa_{\min}} \left(M(\bar{\lambda}, \bar{\mu}, \rho) + \alpha_j \right) \Delta_j,$$

and

$$\begin{aligned} u_i \geq 0 \quad \textit{and} \quad & a_i^T x - b_i \leq 0 && \forall\, i \in \mathcal{I}, \\ a_i^T x - b_i < -r_j \,\|\, a_i \,\| \quad \Rightarrow \quad & u_i = 0 && \forall\, i \in \mathcal{I}, \\ & a_i^T x - b_i = 0 && \forall\, i \in \mathcal{E}. \end{aligned}$$

Note that in the context of GSS methods we do not know the specific values of the v_i and u_i, only that they exist.

It remains to impose a suitable stopping tolerance $\Delta_{\text{tol}}^{(k)}$ for subproblem k in the outer loop of the augmented Lagrangian method. Suppose that we use the linearly constrained GSS Algorithm 4.1 to minimize $L(x; \bar{\lambda}^{(k)}, \bar{\mu}^{(k)}, \rho^{(k)})$. Let $j_*^{(k)}$ be the unsuccessful iteration after which the stopping criterion is satisfied (i.e., $\Delta_{j_*^{(k)}+1} = \theta_{j_*^{(k)}} \Delta_{j_*^{(k)}} < \Delta_{\text{tol}}^{(k)}$). Let $a_{\max} = \max_{i \in \mathcal{I}} \{\| a_i \|\}$. In addition, suppose we use the choice $r_j = \Delta_j$ for all j. Then Proposition 5.3 tells us that (3.1)–(3.4) hold with

$$\varepsilon_1^{(k)} = \frac{1}{\kappa_{\min}} (M(\bar{\lambda}^{(k)}, \bar{\mu}^{(k)}, \rho^{(k)}) + \alpha_{j_*^{(k)}}) \frac{1}{\theta_{j_*^{(k)}}} \Delta_{\text{tol}}^{(k)}, \tag{5.1}$$

$$\varepsilon_2^{(k)} = a_{\max} \frac{1}{\theta_{j_*^{(k)}}} \Delta_{\text{tol}}^{(k)} \tag{5.2}$$

$$\varepsilon_3^{(k)} = 0. \tag{5.3}$$

To ensure convergence of the augmented Lagrangian scheme we must ensure that $\varepsilon^{(k)} = \max\{\varepsilon_1^{(k)}, \varepsilon_2^{(k)}, \varepsilon_3^{(k)}\} \to 0$. We will do so by ensuring that $\Delta_{\text{tol}}^{(k)} \to 0$ under the following condition.

Condition 5.4. *There exists $C \geq 0$ such that for all k,*

$$M(\bar{\lambda}^{(k)}, \bar{\mu}^{(k)}, \rho^{(k)}) \leq C \left(1 + \left\| \bar{\lambda}^{(k)} \right\| + \left\| \bar{\mu}^{(k)} \right\| + \rho^{(k)}\right).$$

Condition 5.4 will hold, for instance, if there exists a compact set on which ∇f, ∇h, and ∇g are Lipschitz continuous and which contains, for all k, the sequence of iterates $\{x_j^{(k)}\}$ produced by the linearly constrained GSS Algorithm 4.1 applied to (2.2).

From (5.1)–(5.3) we see that ensuring that $\varepsilon^{(k)} \to 0$ as $k \to \infty$ requires not just $\Delta_{\text{tol}}^{(k)} \to 0$ but also $(M(\bar{\lambda}^{(k)}, \bar{\mu}^{(k)}, \rho^{(k)}) + \alpha_{j_*^{(k)}}) \Delta_{\text{tol}}^{(k)} \to 0$. Since both $M(\bar{\lambda}^{(k)}, \bar{\mu}^{(k)}, \rho^{(k)})$ and $\alpha_{j_*^{(k)}}$ will grow with $\| \bar{\lambda}^{(k)} \|$, $\| \bar{\mu}^{(k)} \|$, and $\rho^{(k)}$, we must guard against the possibility that any of the latter quantities grows without bound. The augmented Lagrangian framework in Algorithm 3.1 explicitly enforces boundedness of the multiplier estimates $\bar{\lambda}^{(k)}$ and $\bar{\mu}^{(k)}$. However, there remains the possibility that the sequence of penalty parameters $\{\rho^{(k)}\}$ is unbounded. If this occurs, the augmented Lagrangian will become increasingly nonlinear.

As $\bar{\lambda}^{(k)}$, $\bar{\mu}^{(k)}$, and $\rho^{(k)}$ increase in magnitude, the GSS stopping tolerance $\Delta_{\text{tol}}^{(k)}$ for the inner iteration should be tightened to ensure (3.1) holds for a reasonably small $\varepsilon_1^{(k)}$. Consequently, we have the following update rule for the subproblem stopping tolerance. First, choose a function $\delta(\bar{\lambda}, \bar{\mu}, \rho)$ such that $\left(\| \bar{\lambda} \| + \| \bar{\mu} \| + \rho\right) = O(\delta(\bar{\lambda}, \bar{\mu}, \rho))$ as $\left(\| \bar{\lambda} \| + \| \bar{\mu} \| + \rho\right) \to \infty$. The subproblem stopping tolerance can then be updated as follows: Choose $\xi_{\min}$ and $\xi_{\max}$, independent of k, satisfying $0 < \xi_{\min} \leq \xi_{\max} < 1$. Choose $\xi^{(k)}$ satisfying $\xi_{\min} \leq \xi^{(k)} \leq \xi_{\max}$. Set

$$\Delta_{\text{tol}}^{(k+1)} = \xi^{(k)} \Delta_{\text{tol}}^{(k)} / \delta(\bar{\lambda}^{(k+1)}, \bar{\mu}^{(k+1)}, \rho^{(k+1)}).$$

With this update rule, Condition 5.4 and the requirement that $\delta_j \geq \delta_{\min} > 0$ guarantee that $\varepsilon^{(k)} = \max\{\varepsilon_1^{(k)}, \varepsilon_2^{(k)}, \varepsilon_3^{(k)}\} \to 0$ as $\Delta_{\text{tol}}^{(k)} \to 0$. For instance, we could choose $\delta_{\text{tol}} > 0$ and let

$$\delta(\bar{\lambda}, \bar{\mu}, \rho) = \max\{1,\ (1 + \| \bar{\lambda} \| + \| \bar{\mu} \| + \rho)/\delta_{\text{tol}}\}. \tag{5.4}$$

A similar update rule appears in [12]. The threshold trigger δ_{tol} is intended to prevent overly aggressive reductions in $\Delta_{\text{tol}}^{(k)}$. We are indebted to Rakesh Kumar of The

MathWorks for relating to us his experience, from implementing and testing the Matlab GADS Toolbox version of the GSS augmented Lagrangian algorithm from [16], that decreasing $\Delta_{\text{tol}}^{(k+1)}$ too aggressively leads to computational inefficiency. Further discussion of the stopping tolerance update may be found in [12].

Recall that we are working under the assumption that exact derivatives (or sufficiently accurate estimates) of f, g, and h are unavailable; hence, our need to tighten the stopping criterion in reaction to increased nonlinearity of the augmented Lagrangian due to growth in the multipliers or the penalty parameter. The same issue would arise if one were to minimize the augmented Lagrangian using finite differences to estimate the Jacobian of the constraints. In the latter case, the nonlinearity surfaces in the truncation error of the finite difference estimates. If the multipliers or the penalty parameter become large, then the finite difference perturbation used will need to be decreased appreciably in order to control the truncation error and retain assurance that if the finite difference approximation of ∇L satisfies (3.1), then the exact gradient ∇L does as well.

6. Analytical results for the augmented Lagrangian approach.

In [1], there are two sets of results relating to the general version of Algorithm 3.1. The first set of results assumes that the inner iteration always finds an $x^{(k)}$ satisfying (3.1)–(3.4) and then examines global convergence results that use the CPLD constraint qualification. The second set of results considers the boundedness of the penalty parameters. We recap these results in the specific context of our Algorithm 5.1, where the subproblem is linearly constrained.

6.1. Global convergence.

In addition to the assumption that an $x^{(k)}$ satisfying (3.1)–(3.4) can always be found, the results in [1, Section 4] also assume the existence of at least one limit point of the sequence $\{x^{(k)}\}$ generated by Algorithm 3.1.

The first result shows that either a limit point is feasible or it is a KKT point over Ψ of the sum of squares of the infeasibilities for the upper-level constraints. Theorem 6.1 is a special case of the result [1, Theorem 4.1] on which it is based that takes into account that since the lower-level constraints are linear, the CPLD constraint qualification necessarily holds for the lower-level constraints.

THEOREM 6.1. *Let $\{x^{(k)}\}$ be a sequence generated by Algorithm 5.1. Let $x^{(\star)}$ be a limit point of $\{x^{(k)}\}$. Then, if the sequence of penalty parameters $\{\rho^{(k)}\}$ is bounded, the limit point $x^{(\star)}$ is feasible. Otherwise, the limit point $x^{(\star)}$ is a KKT point of the problem*

$$\begin{array}{ll} \textit{minimize} & \frac{1}{2}\left(\| \, h(x) \, \|^2 + \| \max\left(0, g(x)\right) \|^2\right) \\ \textit{subject to} & x \in \Psi. \end{array}$$

The next result says that under the CPLD constraint qualification, feasible limit points are KKT points of (1.1). Moreover, if the stronger Mangasarian-Fromowitz constraint qualification (MFCQ) [18] is also satisfied, then the unprojected multiplier estimates associated with any subsequence of iterates converging to $x^{(\star)}$ are bounded. Let $\Omega = \{\, x \mid h(x) = 0, \ g(x) \leq 0 \,\}$ denote the feasible region with respect to the upper-level constraints.

THEOREM 6.2. *[1, Theorem 4.2] Let $\{x^{(k)}\}_{k\in\mathbb{N}}$ be a sequence generated by Algorithm 5.1. Suppose $x^{(\star)} \in \Omega\cap\Psi$ is a limit point that satisfies the CPLD constraint qualification related to $\Omega\cap\Psi$. Then, $x^{(\star)}$ is a KKT point of the original problem (1.1). Moreover, if $x^{(\star)}$ satisfies the MFCQ and $\{x^{(k)}\}_{k\in K}$ is a subsequence that converges to $x^{(\star)}$, the set $\{\| \, \lambda^{(k+1)} \, \|, \| \, \mu^{(k+1)} \, \|, \| \, v^{(k)} \, \|, \| \, u^{(k)} \, \|\}_{k\in K}$ is bounded.*

6.2. Boundedness of the penalty parameters. As noted in [1], study of conditions under which the penalty parameters are bounded is important since if the penalty parameters are too large, the subproblems tend to be ill-conditioned and thus more difficult to solve. The results in [1, Section 5] are split into two sections: those for equality constraints (only) and those for general constraints. The technique for proving the boundedness of the penalty parameters in the case of general constraints is to reduce (1.1) to a problem consisting only of equality constraints. The equality constraints of the new (reduced) problem are the active constraints at the limit point $x^{(\star)}$. The boundedness of the penalty parameters then follows from the results for problems with equality constraints only. Here we simply recap the result for the general case.

The following assumptions serve as the sufficient conditions for the case of general constraints.

ASSUMPTION 6.3. *[1, Assumption 7] The sequence $\{x^{(k)}\}$ is generated by the application of Algorithm 5.1 to problem (1.1) and $\lim_{k\to\infty} x^{(k)} = x^{(\star)}$.*

ASSUMPTION 6.4. *[1, Assumption 8] The point $x^{(\star)}$ is feasible.*

ASSUMPTION 6.5. *[1, Assumption 9] LICQ holds at $x^{(\star)}$: the set of vectors $\{ \nabla h_i(x^{(\star)}) \mid i = 1, \ldots, m \} \cup \{ \nabla g_i(x^{(\star)}) \mid g_i(x^{(\star)}) = 0 \} \cup \{ a_i \mid i \in \mathcal{E} \} \cup \{ a_i \mid i \in \mathcal{I}, a_i^T x^{(\star)} = b_i \}$ is linearly independent.*

ASSUMPTION 6.6. *[1, Assumption 10] The functions f, h, and g admit continuous second derivatives in a neighborhood of $x^{(\star)}$.*

ASSUMPTION 6.7. *[1, Assumption 11] Define the tangent subspace T as the set of z satisfying*

1. $\nabla h_i(x^{(\star)})^T z = 0, i = 1, \ldots, m,$
2. $\nabla g_i(x^{(\star)})^T z = 0$ *for all i such that $g_i(x^{(\star)}) = 0$, and*
3. $a_i^T z = 0$ *for all $i \in \mathcal{E}$ and $i \in \mathcal{I}$ such that $a_i^T x^{(\star)} = b_i$.*

Then, for all $z \in T$, $z \neq 0$,

$$z^T \left(\nabla^2 f(x^{(\star)}) + \sum_{i=1}^{m} \lambda_i^{(\star)} \nabla^2 h_i(x^{(\star)}) + \sum_{i=1}^{p} \mu_i^{(\star)} \nabla^2 g_i(x^{(\star)}) \right) z > 0.$$

ASSUMPTION 6.8. *[1, Assumption 12] The Lagrange multipliers $\lambda^{(\star)} \in \mathbb{R}^m$ and $\mu^{(\star)} \in \mathbb{R}^p$ satisfy $\bar{\lambda}_{\min} \le \lambda^{(\star)} \le \bar{\lambda}_{\max}$ and $0 \le \mu^{(\star)} \le \bar{\mu}_{\max}$.*

ASSUMPTION 6.9. *[1, Assumption 13] Strict complementarity holds at $x^{(\star)}$ for the inequality constraints $g(x^{(\star)})$.*

The authors note in [1] that Assumption 6.7 is weaker than the usual second-order sufficiency assumption. Assumption 6.9 requires strict complementarity only for the upper-level inequality constraints.

THEOREM 6.10. *[1, Theorem 5.5] Suppose that Assumptions 6.3–6.9 are satisfied and assume that there exists a sequence $\eta^{(k)} \to 0$ such that for all $k \in \mathbb{N}$, $\varepsilon^{(k)} \le \eta^{(k)} \max \left\{ \left\| h(x^{(k)}) \right\|_\infty, \left\| \sigma^{(k)} \right\|_\infty \right\}$. Then the sequence of penalty parameters $\{\rho^{(k)}\}$ is bounded.*

7. Preliminary numerical results. We report numerical results for two sets of problems. In the first set we compare results we obtained with results reported in [1, Section 6.1] for five small test problems chosen because of known pathologies meant to test both the analysis and the comparable performance for IPOPT [22, 24], a well-regarded implementation of an interior-point method. In the second part, we report preliminary results for some problems taken from the CUTEr test suite [9].

In the numerical tests reported in [1, Section 6.1], GENCAN [3], a solver for unconstrained and bound-constrained problems that uses second derivatives and a conjugate gradient preconditioner, was used for the inner iterations. For our tests we used our implementation of a linearly constrained GSS method described in [14, 17].

All tests were run on an Apple MacBook with a 2 GHz Intel Core 2 Duo processor and 1 GB memory running Mac OS X, Version 10.4.11 and using Matlab R2008b.

For the test results reported in Section 7.1, we used the same parameter values for Algorithm 5.1 as those reported in [1, Section 6] so that the results could be compared. Specifically, we initialized the penalty parameter to

$$\rho^{(1)} = \max\left\{10^{-6},\ \min\left\{10,\ \frac{2\,|\,f(x^{(0)})\,|}{\|\,h(x^{(0)})\,\|_2 + \|\,\max(0,\ g(x^{(0)}))\,\|_2}\right\}\right\} \tag{7.1}$$

and used $\tau = 0.5$ and $\gamma = 10$ for testing and updating $\rho^{(k)}$. For the bounds on the multipliers, we used $\bar{\lambda}_{\min} = -10^{20}$ and $\bar{\lambda}_{\max} = \bar{\mu}_{\max} = 10^{20}$; we set $\bar{\lambda}^{(1)} = 0$ and $\bar{\mu}^{(1)} = 0$. For all k, $\varepsilon^{(k)} = 10^{-4}$. Algorithm 5.1 was terminated when

$$\max\left\{\left\|\,h(x^{(k)})\,\right\|_\infty,\ \left\|\,\sigma^{(k)}\,\right\|_\infty\right\} \le 10^{-4}. \tag{7.2}$$

The definition of $\sigma^{(k)}$ given in (3.5) ensures that if $\|\,\sigma^{(k)}\,\|_\infty \le 10^{-4}$, then $g_i(x^{(k)}) \le 10^{-4}$ for all $i \in \{1, \ldots, p\}$ and $\mu_i^{(k)} = 0$ whenever $g_i(x^{(k)}) < 10^{-4}$.

For the results reported in Section 7.2 we made two changes to the values of the algorithmic parameters. First, we replaced the value 10^{-6} in (7.1) with the value 1.0 since in our preliminary experiments we found that the value $\rho^{(1)} = 10^{-6}$ (as was often the case) put too little weight on achieving feasibility in the initial rounds of the search and thus triggered more outer iterations than were necessary to achieve acceptable solutions. Second, we replaced $\pm 10^{20}$ with $\pm$`Inf` as bounds on the multipliers since Matlab gracefully handles the unambiguous floating point infinity.

7.1. Tests of the convergence analysis. Here we compare results with those given in the section of the same name in [1, Section 6.1]. We repeat the experiments on these simple but pathological problems for the following reason. In the tests done in [1] using the augmented Lagrangian algorithm Algorithm 3.1, the tolerance parameters $\varepsilon^{(k)}$ are monotonically decreasing. In the GSS adaptation Algorithm 5.1, we are not guaranteed monotonic decrease. It is natural to wonder whether this difference would cause different behavior on pathological problems, or whether if we could solve the subproblems to sufficient accuracy, we would attain equivalent results. The latter turns out to be the case.

Example 1. Convergence to KKT points that do not satisfy MFCQ. In this problem, no feasible point satisfies MFCQ, but all feasible points satisfy CPLD:

$$\begin{array}{ll} \text{minimize} & x_1 \\ \text{subject to} & x_1^2 + x_2^2 - 1 \le 0 \\ & -x_1^2 - x_2^2 + 1 \le 0. \end{array}$$

The global solution is $x^{(\star)} = (-1, 0)^T$.

In [1], Algorithm 3.1 was tested using 100 randomly generated starting points in the square $[-10, 10]^2$ (all 100 trials converged to the global solution), but specific statistics are given only for the starting point $(5, 5)^T$. Table 7.1 shows that from the same starting point we achieved comparable results using linearly constrained GSS to solve the subproblems, at least in terms of the final results and the number of outer augmented Lagrangian iterations required.

inner method	# it.	initial ρ	final ρ	multiplier estimates
GENCAN	14	4.1649E-03	4.1649E-01	4.998E-01 0.0000E+00
GSS	14	4.1649E-03	4.1649E-01	5.000E-01 0.0000E+00

TABLE 7.1
Results for Example 1.

Example 2. Convergence to a non-KKT point. For the following problem, the constraints are linearly dependent for all $x \in \mathbb{R}$. In spite of the this, the only point that satisfies the conditions of Theorem 6.1 is $x^{(\star)} = 0$:

$$\begin{array}{ll} \text{minimize} & x \\ \text{subject to} & x^2 = x^3 = x^4 = 0. \end{array}$$

In [1], Algorithm 3.1 was tested using 100 randomly generated starting points in the interval $[-10, 10]$ (all 100 trials converged to the global solution), but specific statistics are given only for the starting point 5. Table 7.2 shows that from the same starting point we again achieved comparable results.

inner method	# it.	initial ρ	final ρ	multiplier estimates
GENCAN	20	2.4578e-05	2.4578e+05	5.2855e+01 -2.0317e+00 4.6041e-01
GSS	20	2.4578e-05	2.4578e+05	5.2881e+01 -2.0322e+00 4.6040e-01

TABLE 7.2
Results for Example 2.

Example 3. Infeasible stationary points [5, 10]. The following problem has a global solution at $x^{(\star)} = (0, 0)^T$, but it also has a stationary point for the infeasibility at $x = (0.5, \sqrt{0.5})^T$.

$$\begin{array}{ll} \text{minimize} & 100(x_2 - x_1^2)^2 + (x_1 - 1)^2 \\ \text{subject to} & x_1 - x_2^2 \le 0 \\ & x_2 - x_1^2 \le 0 \\ & -0.5 \le x_1 \le 0.5 \\ & x_2 \le 1. \end{array}$$

In [1], Algorithm 3.1 was tested using 100 randomly generated starting points in the square $[-10, 10]^2$ (all 100 trials converged to the global solution), but specific statistics are given only for the starting point $(5, 5)^T$. As can be seen in Table 7.3, from the same starting point we achieved comparable results. (Note that we report the starting value for the penalty parameter given in [1], though the value we used for $\rho^{(1)}$ was obtained from (7.1), the formula given in [1].)

Example 4. Difficult-for-barrier [4, 5, 22]. This problem was selected due to the observation made in [5] that although the problem is well-posed, many barrier-SQP methods fail to obtain feasibility for a range of infeasible starting points:

$$\begin{array}{ll} \text{minimize} & x_1 \\ \text{subject to} & x_1^2 - x_2^2 + a = 0 \\ & x_1 - x_3 - b = 0 \\ & 0 \le x_2 \\ & 0 \le x_3. \end{array}$$

inner method	# it.	initial ρ	final ρ	multiplier estimates
GENCAN	6	1.0000E+00	1.0000E+01	1.9998E+00 3.3390E-03
GSS	6	1.0000E+01	1.0000E+01	1.9999E+00 4.4878E-03

TABLE 7.3
Results for Example 3.

Following the suggestion in [5], Algorithm 3.1 was tested in [1] using two different values of the parameters a and b and the corresponding initial points. We tested the choice $(a,b) = (-1, 0.5)$ and $x^{(0)} = (-2, 1, 1)^T$ since in this instance IPOPT was reported to have stopped upon indicating convergence to a stationary point for infeasibility. As can be seen in Table 7.4, from the same starting point we achieved comparable results and converged to the global solution $x^{(\star)} = (1, 0, 0.5)^T$. (Once again we report the starting value for the penalty parameter given in [1], though the value we used for $\rho^{(1)}$ was obtained from (7.1), the formula given in [1].)

inner method	# it.	initial ρ	final ρ	multiplier estimates
GENCAN	5	2.4615E+00	2.4615E+00	-5.0001E-01 -1.3664E-16
GSS	5	2.4615E-01	2.4615E+01	-4.9981e-01 3.0048e-04

TABLE 7.4
Results for Example 4.

Example 5. Preference for global minimizers. The following problem has 2^n local minimizers corresponding to the vertices of the hypercube $[-1,1]^n$, but a unique global minimizer at $x^{(\star)} = (-1,\ldots,-1)^T$.

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^n x_i \\ \text{subject to} & x_i^2 = 1, \quad i = 1,\ldots,n. \end{array}$$

Following the lead in [1], we set $n = 100$. We ran from ten different starting points using the Matlab Mersenne Twister pseudorandom number generator to generate uniform random values in the hypercube $[-100, 100]^n$. The qualitative behavior across all ten runs was the same. In all cases we found the global solution with $\max(x) \in [\texttt{-9.9997e-01}, \texttt{-9.9996e-01}]$ and $f(x) = \texttt{-1.0000E+02}$. It took either 19 or 20 outer iterations to satisfy the stopping condition based on the maximum constraint violation. At the start, $\rho = \texttt{1.0E-06}$ for all ten runs (the norm of the constraint violation at the starting point was $\approx$ 1.0E+04 while the absolute value of the objective at the starting point was $\approx$ 1.0E+02. Upon termination, $\rho = \texttt{1.0E+00}$ when finished in 19 outer iterations and $\rho = \texttt{1.0E+01}$ when finished in 20 outer iterations. Across all ten runs, $\min(\lambda) \in [\texttt{0.49938}, \texttt{0.49993}]$ and $\max(\lambda) \in [\texttt{0.50005}, \texttt{0.50061}]$.

In [1], the authors report executing 100 runs using the function `Uniform01` provided by AMPL [8] to generate uniform random initial points in the hypercube $[-100, 100]^n$. They report that their algorithm converged to the global solution in all cases while IPOPT never found the global solution. When starting from the first random point, they note that 4 outer iterations were required to satisfy the stopping condition, that the final penalty parameter was 5.0882E+00 (the initial one was 5.0882E-01), and that the final multipliers were all equal to 4.9999E-01.

When we tried one run that restricted the range of the starting values to the unit hypercube $[-1, 1]^n$, the value computed for the initial ρ using (7.1) was `2.031458e-01`, which is at least of the same order of magnitude as that reported in [1]. Algorithm 5.1 then converged to the global solution in only 6 outer iterations with a final value for ρ of `2.031458e+00`. Since f is a linear function and there are no bounds on the variables, regardless of the starting point the degree to which the upper-level constraint violations are penalized plays a significant role, especially in the progress made during the first outer iteration.

7.2. CUTEr **problems.** We report results for a selection of problems from the CUTEr test suite [9]. Basic characteristics of the problems we used are given in Table 7.5. The problems selected exhibit a wide range of features. Objective functions are a mix of linear ("`L`" in the CUTEr classification scheme), quadratic ("`Q`"), sum-of-squares ("`S`"), and general nonlinear ("`O`"). All problems have bounds on at least some of the variables (though only a few have lower and upper bounds on all of the variables), thus justifying the use of specialized software to handle the bounds as lower-level constraints. Some problems have linear constraints, which we also treat explicitly as lower-level constraints. We focused on problems classified as "real" ("`-R`") and "model" ("`-M`"), as opposed to "academic" ("`-A`"), as being more indicative of the types of problems to which derivative-free methods are applied.

problem	n	# of nonlinear eq.	# of nonlinear ineq.	# of linear eq.	# of linear ineq.	# of bounds lower	# of bounds upper	CUTEr classification
AIRPORT	84	0	42	0	0	84	84	`SQR2-MN-84-42`
CANTILVR	5	0	1	0	0	5	0	`LOR2-MN-5-1`
CRESC4	6	0	8	0	0	4	1	`OOR2-MY-6-8`
CRESC50	6	0	100	0	0	4	1	`OOR2-MY-6-100`
CRESC100	6	0	200	0	0	4	1	`OOR2-MY-6-200`
DECONVC	61	1	0	0	0	51	0	`SQR2-MN-61-1`
DNIEPER	61	24	0	0	0	56	56	`QOR2-MN-61-24`
HS107	9	6	0	0	0	5	3	`OOR2-MY-9-6`
HS114	10	2	4	1	4	10	10	`QOR2-MY-10-11`
HS69	4	2	0	0	0	4	4	`OOR2-MN-4-2`
PRODPL0	60	0	4	20	5	60	0	`LQR2-RY-60-29`
PRODPL1	60	0	4	20	5	60	0	`LQR2-RY-60-29`
ROBOT	14	2	0	0	0	7	7	`QOR2-MY-14-2`
TWOBARS	2	0	2	0	0	2	2	`OOR2-MN-2-2`

TABLE 7.5
Summary of problem characteristics.

7.2.1. Starting values for the results reported here. For all problems we started with the value of $x^{(0)}$ specified by CUTEr. Our software for handling the lower-level constraints requires that all iterates be feasible with respect to the linear constraints. The $x^{(0)}$ given in the CUTEr `.SIF` file for all the problems in Table 7.5 is feasible with respect to the bounds on the variables. However, for those problems for which there are linear constraints, $x^{(0)}$ was *infeasible* with respect to the linear constraints, as shown in Table 7.6. For these three problems, we projected the $x^{(0)}$ given by CUTEr into Ψ to produce a new $x^{(0)}$ that is feasible to within an acceptable tolerance, also shown in Table 7.6. Worth noting is that for all three of these problems, multiple times during the course of the inner iterations, the search encountered working sets of linear constraints that had a degenerate vertex at the origin. The latter situation requires special techniques to compute the set of search directions, as discussed in [14, Section 5.4.2].

problem	max linear constraint bound violation			
	before projection into $\boldsymbol{\Psi}$		after projection into $\boldsymbol{\Psi}$	
	lower	upper	lower	upper
HS114	4.4000e-01	0.0000e+00	0.0000e+00	4.5475e-13
PRODPL0	3.3330e+00	0.0000e+00	1.7764e-15	8.8818e-16
PRODPL1	3.3330e+00	0.0000e+00	1.7764e-15	8.8818e-16

TABLE 7.6

Infeasibility of the CUTEr *starting point $x^{(0)}$ with respect to the linear constraints.*

In Table 7.7 we give the value of $f(x^{(0)})$ for $x^{(0)} \in \Psi$, the norm of the nonlinear constraint violations at $x^{(0)} \in \Psi$, and the "best known" objective value $f_{\rm lo}$. In many

problem	$f(x^{(0)})$	$\| h(x^{(0)}) \|_\infty$	$\| g(x^{(0)}) \|_\infty$	$f_{\rm lo}$
AIRPORT	0.0000e+00	—	1.0360e+02	4.7953e+04
CANTILVR	3.1200e-01	—	1.2400e+02	1.3400e+00
CRESC4	2.8822e+00	—	1.7153e+03	3.9244e+00
CRESC50	2.8822e+00	—	1.7118e+03	1.9805e+00
CRESC100	2.8822e+00	—	1.7073e+03	6.9621e+01
DECONVC	1.1035e+02	9.9570e-01	—	2.5695e-03
DNIEPER	-8.9102e+02	1.5462e+01	—	1.8744e+04
HS107	4.8533e+03	8.0000e-01	—	5.0550e+03
HS114	-8.7435e+02	8.9467e-02	0.0000e+00	-1.7688e+03
HS69	-6.3135e+02	6.8269e-01	—	-9.5671e+02
PRODPL0	7.5179e+01	—	2.9029e+00	5.8790e+01
PRODPL1	7.5179e+01	—	2.1647e+00	3.5739e+01
ROBOT	0.0000e+00	4.0000e+00	—	5.4628e+00
TWOBARS	1.4142e+00	—	5.7826e-01	1.5086e+00

TABLE 7.7

Starting values for the objective at $x^{(0)} \in \Psi$, max norms of the constraint violations at $x^{(0)} \in \Psi$, and best known optimal feasible values.

instances, this last value is given in the `.SIF` file that defines the problem. If not, we used the value reported in [23]. We qualify "best known" because "best" may well depend on the amount of infeasibility allowed or tolerated upon termination, and with respect to which constraints. While the CUTEr `.SIF` files often contain a best known ("`LO SOLTN`") value, we have not found one that contains any information on the norm of the constraint violations at the point $x_{\rm lo}$ that produced the value reported for $f_{\rm lo}$. The norms of the constraint violations reported in [23] vary widely. For instance, the CUTEr `.SIF` files for the problems CRESC4, CRESC50, and CRESC100 do not contain best known values. The values for $f_{\rm lo}$ given in [23] for these three problems have associated constraint violation norms `3.2E+00`, `6.6E+03`, and `1.5E+04`, respectively (where the violations are respect to *all* constraints, including bounds on the variables). Yet for these three problems we obtained both significantly lower objective values and significantly lower constraint violations, as can be seen in Tables 7.8–7.10.

Because we terminated the outer iterations when $\Delta_{\rm tol}^{(k)}$ fell below 10^{-4} (analogous to the criterion $\varepsilon^{(k)} < 10^{-4}$ in [1]), we report no more than five significant decimal digits. The format used for the `.SIF` files supports no more than seven significant decimal digits in the problem specification. We have found that in some `.SIF` files the data defining the problem are truncated to only two or three decimal digits. Thus, demanding full double-precision accuracy is unrealistic.

7.2.2. Results when the linear constraints are treated explicitly. Tables 7.8–7.12 summarize the results we obtained. From [17, Section 6.2] we borrowed a measure which we call *fraction of optimal improvement* to show how much progress was made toward the best known optimal solution, given the starting values reported in Table 7.7. Specifically, let `f0` be the value $f(x^{(0)})$ for $x^{(0)} \in \Psi$. Let `f_sol` be the best objective value found by our method, given the stopping condition imposed (e.g., the value of f at termination reported in Table 7.8). Let `f_lo` be the "best known" value of the objective (e.g., the value f_{lo} given in Table 7.7). Then fraction of optimal improvement, with respect to the value of the objective function, is defined as

$$\frac{|\texttt{f0} - \texttt{f\_sol}|}{|\texttt{f0} - \texttt{f\_lo}|}.$$

Note that we do not report fraction of optimal improvement for the problems CRESC4, CRESC50, and CRESC100 since we obtained significantly better solutions than those reported for IPOPT and, as already noted, no values are given in the CUTEr .SIF files for these three problems.

problem	value of f at termination	frac. of opt. improvement
AIRPORT	4.7953e+04	1.0000e+00
CANTILVR	1.3400e+00	1.0000e+00
CRESC4	8.7215e-01	—
CRESC50	8.0824e-01	—
CRESC100	6.4331e-01	—
DECONVC	2.6352e-06	1.0000e+00
DNIEPER	1.8755e+04	1.0006e+00
HS107	5.0546e+03	9.9794e-01
HS114	-1.7677e+03	9.9877e-01
HS69	-9.5669e+02	9.9992e-01
PRODPL0	5.8790e+01	9.9996e-01
PRODPL1	3.5740e+01	9.9997e-01
ROBOT	5.4628e+00	1.0000e+00
TWOBARS	1.5087e+00	1.0002e+00

TABLE 7.8
Summary of improvements in the objective values.

Equivalent definitions hold for the fraction of optimal improvement with respect to the norm of the nonlinear constraint violations, though for these definitions there is no ambiguity with respect to the optimal value, since it is zero. We include these values in Table 7.9.

In Table 7.10 we show the maximum constraint violation, with respect to both the bounds on the variables and the linear constraints, at the solution we obtained. Any violations are small, as one would expect, since we enforce feasibility with respect to these constraints (with a tolerance to account for floating-point error). In Table 7.11 we give the number of outer iterations, as well as the initial and final value of the penalty parameter. Finally, in Table 7.12 we give the minimum and maximum values of λ_i and μ_i upon termination. The purpose is to show that for these problems neither the penalty parameters nor the multipliers grew excessively large in magnitude.

7.2.3. Results when the linear constraints are folded into the augmented Lagrangian. We are not aware of any published results showing whether there is an advantage to treating linear constraints explicitly in the context of an augmented Lagrangian approach to solving nonlinear programming problems. We were therefore curious how our results for the problems listed in Table 7.6 would change if we folded the linear constraints into the augmented Lagrangian, rather than dealing

problem	$\| h(x) \|_\infty$ at termination	frac. of opt. improvement	$\| g(x) \|_\infty$ at termination	frac. of opt. improvement
AIRPORT	—	—	1.9708e-05	1.0000e+00
CANTILVR	—	—	0.0000e+00	1.0000e+00
CRESC4	—	—	2.9285e-05	1.0000e+00
CRESC50	—	—	0.0000e+00	1.0000e+00
CRESC100	—	—	4.0980e-06	1.0000e+00
DECONVC	2.9241e-05	9.9997e-01	—	—
DNIEPER	7.3508e-05	1.0000e+00	—	—
HS107	5.7909e-05	9.9993e-01	—	—
HS114	7.4444e-05	9.9917e-01	0.0000e+00	1.0000e+00
HS69	3.4658e-06	9.9999e-01	—	—
PRODPL0	—	—	7.1451e-06	1.0000e+00
PRODPL1	—	—	8.7444e-06	1.0000e+00
ROBOT	3.0952e-05	9.9999e-01	—	—
TWOBARS	—	—	0.0000e+00	1.0000e+00

TABLE 7.9

Summary of improvements in the feasibilities for the nonlinear constraints.

problem	maximum constraint violation at solution			
	bounds on x		bounds on Ax	
	lower	upper	lower	upper
AIRPORT	0.0000e+00	0.0000e+00	—	—
CANTILVR	0.0000e+00	0.0000e+00	—	—
CRESC4	0.0000e+00	0.0000e+00	—	—
CRESC50	0.0000e+00	0.0000e+00	—	—
CRESC100	0.0000e+00	0.0000e+00	—	—
DECONVC	0.0000e+00	0.0000e+00	—	—
DNIEPER	0.0000e+00	0.0000e+00	—	—
HS107	0.0000e+00	0.0000e+00	—	—
HS114	0.0000e+00	0.0000e+00	0.0000e+00	4.5475e-12
HS69	0.0000e+00	0.0000e+00	—	—
PRODPL0	1.0760e-15	0.0000e+00	1.9984e-15	1.3323e-15
PRODPL1	9.5104e-16	0.0000e+00	2.6645e-15	2.6645e-15
ROBOT	0.0000e+00	0.0000e+00	—	—
TWOBARS	0.0000e+00	0.0000e+00	—	—

TABLE 7.10

Summary of feasibility of the linear constraints at termination.

with them explicitly as in the results reported in Section 7.2.2. The results of our second round of tests are given in Tables 7.13–7.14.

These results suggest that the explicit treatment of the linear constraints is advantageous, as one would expect. Only one of the three tests, that for PRODPL0, terminated because the stopping criterion (7.2) for Algorithm 5.1 was satisfied. In the other two instances, the test terminated because the penalty parameter reached the value 10^{20}, a clear indication that the search had stalled. Even for the problem PRODPL0, the solution obtained is arguably not as good as that obtained when the linear constraints were handled directly. We conjecture that the difficulties encountered when including the linear constraints in the augmented Lagrangian is due to degeneracy of the linear constraints.

8. Conclusions. The work described here combines derivative-free generating set search methods for linearly constrained optimization and the general augmented Lagrangian framework of [1]. The approach inherits the favorable analytical properties (convergence and boundedness of penalty parameters) of the underlying augmented

problem	penalty parameter initial	penalty parameter at termination	# of outer iterations
AIRPORT	1.0000e+00	1.0000e+08	17
CANTILVR	1.0000e+00	1.0000e+04	9
CRESC4	1.0000e+00	1.0000e+04	9
CRESC50	1.0000e+00	1.0000e+05	10
CRESC100	1.0000e+00	1.0000e+04	9
DECONVC	1.0000e+01	1.0000e+02	3
DNIEPER	1.0000e+00	1.0000e+07	18
HS107	1.0000e+01	1.0000e+05	15
HS114	1.0000e+01	1.0000e+06	12
HS69	1.0000e+01	1.0000e+04	10
PRODPL0	1.0000e+01	1.0000e+06	11
PRODPL1	1.0000e+01	1.0000e+06	12
ROBOT	1.0000e+00	1.0000e+01	6
TWOBARS	8.4585e+00	8.4585e+04	9

TABLE 7.11
Summary of penalty parameters and number of outer iterations.

problem	λ min	λ max	μ min	μ max
AIRPORT	—	—	0.0000e+00	2.8776e+03
CANTILVR	—	—	4.4060e-01	4.4060e-01
CRESC4	—	—	0.0000e+00	4.6755e-01
CRESC50	—	—	-2.6968e-02	0.0000e+00
CRESC100	—	—	-9.4808e-01	6.8421e-01
DECONVC	-2.0521e-04	-2.0521e-04	—	—
DNIEPER	-3.9246e+03	3.8863e+03	—	—
HS107	-1.4789e+00	5.2138e+03	—	—
HS114	-3.9894e+01	3.8048e+01	0.0000e+00	0.0000e+00
HS69	-3.4422e+01	4.3971e+01	—	—
PRODPL0	—	—	9.5053e+00	1.2994e+01
PRODPL1	—	—	0.0000e+00	4.6803e+01
ROBOT	-1.0641e-01	2.3434e+00	—	—
TWOBARS	—	—	0.0000e+00	1.4904e+00

TABLE 7.12
Summary of multipliers obtained.

Lagrangian framework. Moreover, the preliminary numerical results for this approach are promising. Our numerical results also help confirm that explicit treatment of linear constraints is beneficial. More work remains to be done on the implementation, however, and our experience implementing GSS for linearly constrained problems taught us that an effective implementation requires a great deal of care and experimentation.

Acknowledgments. We thank Michael Friedlander, Philip Gill, Michael Saunders, and Margaret Wright for lively and helpful discussions at the 20th International Symposium on Mathematical Programming (ISMP 2009) on the topic of treating linear constraints directly in the implementation of an augmented Lagrangian method.

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 (2009), pp. 1286–1309.

[2] D. P. BERTSEKAS, *Constrained Optimization and Lagrange Multiplier Methods*, Academic Press, New York, 1982.

problem	value of f at termination	frac. of opt. improvement
HS114	-1.6403e+03	8.5663e-01
PRODPL0	6.0647e+01	1.0316e+00
PRODPL1	4.2852e+01	1.1990e+00

TABLE 7.13

Summary of improvements in the objective values when the linear constraints are folded into the augmented Lagrangian.

problem	$\Vert h(x) \Vert_\infty$ at termination	frac. of opt. improvement	$\Vert g(x) \Vert_\infty$ at termination	frac. of opt. improvement
HS114	2.0383e-02	9.5367e-01	7.1419e-05	1.0000e+00
PRODPL0	7.8125e-06	1.0000e+00	0.0000e+00	1.0000e+00
PRODPL1	7.8125e-06	1.0000e+00	3.6621e-04	1.0000e+00

TABLE 7.14

Summary of improvements in the feasibilities for the linear and nonlinear constraints.

[3] E. G. Birgin and J. M. Martínez, *Large-scale active-set box-constrained optimization method with spectral projected gradients*, Computational Optimization and Applications, 23 (2002), pp. 101–125.

[4] R. H. Byrd, M. Marazzi, and J. Nocedal, *On the convergence of Newton iterations to non-stationary points*, Mathematical Programming, 99 (2004), pp. 127–148.

[5] L. Chen and D. Goldfarb, *Interior-point ℓ_2-penalty methods for nonlinear programming with strong global convergence properties*, Mathematical Programming, 108 (2006), pp. 1–36.

[6] A. R. Conn, N. Gould, A. Sartenaer, and P. L. Toint, *Convergence properties of an augmented Lagrangian algorithm for optimization with a combination of general equality and linear constraints*, SIAM Journal on Optimization, 6 (1996), pp. 674–703.

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

[8] R. Fourer, D. M. Gay, and B. W. Kernighan, *AMPL: A Modeling Language for Mathematical Programming*, Thomson/Brooks/Cole, Pacific Grove, California, second ed., 2003.

[9] N. I. M. Gould, D. Orban, and P. L. Toint, *CUTEr (and SifDec), a constrained and unconstrained testing environment, revisited*, ACM Trans. Math. Software, 29 (2003), pp. 373–394. See also http://cuter.rl.ac.uk/cuter-www/.

[10] W. Hock and K. Schittkowski, *Test examples for nonlinear programming codes*, in Lecture Notes in Economics and Mathematical Systems #187, Springer-Verlag, New York, 1981.

[11] T. G. Kolda, R. M. Lewis, and V. Torczon, *Optimization by direct search: New perspectives on some classical and modern methods*, SIAM Review, 45 (2003), pp. 385–482.

[12] ———, *A generating set direct search augmented Lagrangian algorithm for optimization with a combination of general and linear constraints*, Tech. Rep. SAND2006-5315, Sandia National Laboratories, Albuquerque, NM 87185 and Livermore, CA 94550, August 2006.

[13] ———, *Stationarity results for generating set search for linearly constrained optimization*, SIAM J. Optim., 17 (2006), pp. 943–968.

[14] R. M. Lewis, A. Shepherd, and V. Torczon, *Implementing generating set search methods for linearly constrained minimization*, SIAM J. Sci. Comput., 29 (2007), pp. 2507–2530.

[15] R. M. Lewis and V. Torczon, *Pattern search methods for linearly constrained minimization*, SIAM J. Optim., 10 (2000), pp. 917–941.

[16] ———, *A globally convergent augmented Lagrangian pattern search algorithm for optimization with general constraints and simple bounds*, SIAM J. Optim., 12 (2002), pp. 1075–1089.

[17] ———, *Active set identification for linearly constrained minimization without explicit derivatives*, SIAM Journal on Optimization, 20 (2009), pp. 1378–1405.

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

[19] J.-J. Moreau, *Décomposition orthogonale d'un espace hilbertien selon deux cônes mutuellement polaires*, Comptes Rendus de l'Académie des Sciences de Paris, 255 (1962), pp. 238–240.

[20] L. Qi and Z. Wei, *On the constant positive linear dependence condition and its applicaton to*

SQP methods, SIAM Journal on Optimization, 10 (2000), pp. 963–981.

[21] The Mathworks, Inc., *Genetic algorithm and direct search toolbox user's guide, version 2.0*. http://www.mathworks.com/products/gads/, January 2005.

[22] A. Wächter and L. T. Biegler, *Failure of global convergence for a class of interior point methods for nonlinear programming*, Mathematical Programming, 88 (2000), pp. 565–574.

[23] ———, *Table of results for the CUTEr test set for IPOPT*. Available at www.research.ibm.com/people/a/andreasw/papers/Ipopt-table.ps, January 2004.

[24] ———, *On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming*, Mathematical Programming, 106 (2006), pp. 25–57.