A RECURSIVE QUADRATIC PROGRAMMING ALGORITHM FOR SEMI-INFINITE OPTIMIZATION PROBLEMS

by

E. Polak and A. L. Tits

Memorandum No. UCB/ERL M80/50

22 September 1980

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

A RECURSIVE QUADRATIC PROGRAMMING ALGORITHM FOR SEMI-INFINITE OPTIMIZATION PROBLEMS

E. Polak and A. L. Tits

Abstract

The well known, local recursive quadratic programming method introduced by E. R. Wilson is extended to apply to optimization problems with constraints of the type $\max_{\omega} \phi(x,\omega) \leq 0$, where ω ranges over a compact interval of the real line. A scheme is proposed , which results in a globally convergent conceptual algorithm. Finally, two implementable versions are presented both of which converge quadratically.

Keywords: nonlinear programming, control system design, recursive quadratic programming, global convergence.

---

[*] Research sponsored by the National Science Foundation Grant ECS-79-13148 and the Air Force Office of Scientific Research (AFOSR) United States Air Force Contract No. F49620-79-C-0178.

1. Introduction.

This paper is concerned with a particular kind of semi-infinite optimization problem which frequently arises in control system design [1,2,3,4,5]. The problem is characterized by inequalities of the form $\phi(x,\omega) \leq 0$ for all $\omega \in \Omega \subset \mathbb{R}$, where $x \in \mathbb{R}^n$ is the design vector, $\omega \in \mathbb{R}$ is a frequency or time parameter and Ω is a compact interval. A key property of ϕ which holds true in the case of control system design, is that $\phi(x,\cdot)$ has only a finite number of local maxima in Ω. This fact has been strongly utilized in the first order algorithms for these problems, see [3,4]. Recently, it has been shown in [5] that this property can also be used to construct a Newton method, for solving inequalities of the form $\phi(x,\omega) \leq 0$, $\omega \in \Omega$, which solves only finite quadratic programs at each iteration.

The present paper takes up on the ideas in [6,3,4,5] to construct a recursive quadratic programming algorithm for this class of problems. Recursive quadratic programming algorithms were introduced by E. R. Wilson [7], have been analyzed and extended in a large number of subsequent papers by other authors [8,9,10,11,12,13,14], and currently enjoy great popularity.

In their simplest form, recursive quadratic programming algorithms are only locally convergent [8]. A number of schemes have been proposed to stabilize them, i.e. extend their region of convergence [9,12,6]. Some of these schemes use exact penalty functions for testing descent [9,12] in choosing step size; others, e.g. [6], obtain stabilization by cross coupling the R. Q. P. method with a more robust method as specified in the general theory presented in [15]. In this paper, we follow the latter approach. As a result, we obtain two globally convergent versions of a quadratically convergent algorithm for semi-infinte optimization, which solve only finite quadratic programs at each iteration. Furthermore, the cardinality of these programs is bounded and will, usually, be small.

2. Problem Definition and Preliminaries.

We are interested in solving the problem

$$\min\{f(x) \mid g(x) \leq 0,\ \phi(x,\omega) \leq 0\ \forall\omega \in \Omega\} \tag{2.1}$$

with $f : \mathbb{R}^n \to \mathbb{R}$, $g : \mathbb{R}^n \to \mathbb{R}^m$, $\phi : \mathbb{R}^n \times \Omega \to \mathbb{R}^p$, and Ω a compact interval of $\mathbb{R}$. We use the notation

$$\zeta^j(x) = \max_{\omega\in\Omega} \phi^j(x,\omega)$$

$$\zeta(x) = \begin{bmatrix} \zeta^1(x) \\ \vdots \\ \zeta^p(x) \end{bmatrix}$$

and, for any integer k, $\underline{k} \triangleq \{1,2,..,k\}$. Throughout this paper we shall use the following

Assumption 2.1. The functions f, g and ϕ are three times continuously differentiable in all the arguments. ¤

Let $\psi : \mathbb{R}^n \to \mathbb{R}$ be defined by

$$\psi(x) \triangleq \max\{\zeta^j(x),\ j \in \underline{p};\ g^j(x),\ j \in \underline{m}\} \tag{2.2}$$

and, for $a \in \mathbb{R}$, let

$$a_+ \triangleq \max(0,a)$$

The feasible set F is defined by

$$F \triangleq \{x | \psi(x) \leq 0\} \tag{2.3}$$

so that $x \in F$ if and only if $\psi(x)_+ = 0$. For $\varepsilon \geq 0$, the ε-most-active constraints are specified by the sets $J_\varepsilon(x)$ and $\Omega^j_\varepsilon(x)$, $j \in \underline{p}$, defined by

$$J_\varepsilon(x) \triangleq \{j \in \underline{m} | g^j(x) \geq \psi(x)_+ - \varepsilon\} \tag{2.4}$$

$$\Omega^j_\varepsilon(z) \triangleq \{\omega \in \Omega | \phi^j(x,\omega) \geq \psi(x)_+ - \varepsilon\} \ , \ j \in \underline{p} \quad . \tag{2.5}$$

We finally define $\tilde{\Omega}^j_\varepsilon(x)$, the set of ε-most-active local maximizers by

$$\Omega^j_\varepsilon(x) \triangleq \{\omega \in \Omega^j_\varepsilon(x) | \omega \text{ is a local maximizer of } \phi(x,\cdot) \text{ over } \Omega\} \quad .(2.6)$$

Now let $\varepsilon > 0$ be given(possible infinite)and let $x \in \mathbb{R}^n$ be such that

(i) $\tilde{\Omega}^j_\varepsilon(x)$ is a finite set and $\forall\ \omega \in \tilde{\Omega}^j_\varepsilon(x)$, $\phi(x,\omega) + \varepsilon \neq \psi(x)_+$

(ii) for any $\omega \in \tilde{\Omega}^j_\varepsilon(x) \cap \text{int}\ \Omega$, $j \in \underline{p}$ we have

$$\phi^j_{\omega\omega}(x,\omega) \neq 0 \text{ (i.e., } > 0) \tag{2.7}$$

(iii) for any $\omega \in \tilde{\Omega}^j_\varepsilon(x) \cap \text{bd}\ \Omega$, $j \in \underline{p}$ we have

$$\phi^j_\omega(x,\omega) \neq 0 \tag{2.8}$$

(i.e. $\phi^j_\omega(x,\omega) > 0$ if $\omega = \inf \Omega$

< 0 if $\omega = \sup \Omega$) .

Under these conditions, invoking the Implicit Function Theorem for $\phi^j_\omega(x,\omega) = 0$, we see that there exists a ball $B_x = \{\xi | \|\xi - x\| < \rho\}$ (of radius ρ) such that for each $\omega \in \tilde{\Omega}^j_\varepsilon(x)$ there exists a unique, continuously differentiable function $w^j_{x,\omega} : B_x \to \Omega$ such that $w^j_{x,\omega}(\xi) \in \tilde{\Omega}^j_\varepsilon(\xi)$ for all $\xi \in B_x$ and $w_{x,\omega}(x) = \omega$. Its derivative is identically zero if $\omega \in \text{bd}\ \Omega$ (by (iii)) and, if $\omega \in \text{int}\ \Omega$

$$\frac{\partial}{\partial \xi} w^j_{x,\omega}(x) = -\phi^j_{\omega\omega}(x,\omega)^{-1}\phi^j_{\omega x}(x,\omega) \quad . \tag{2.9}$$

For the same given x and each $\omega \in \tilde{\Omega}^j_\varepsilon(x)$ we can then define $\tilde{\eta}^j_{x,\omega} : B_x \to \Omega$ by

$$\tilde{\eta}^j_{x,\omega}(\cdot) = \phi^j(\cdot, w_{x,\omega}(\cdot)) \quad . \tag{2.10}$$

Obviously, if a solution x* of problem (2.1) is such that (i), (ii) and (iii) are satisfied, then the condition

$$\phi^j(x,\omega) \leq 0 \ \forall\, \omega \in \Omega$$

can be replaced by

$$\tilde{\eta}^j_{x^*,\omega}(x) \leq 0 \quad \forall\, \omega \in \tilde{\Omega}^j_\varepsilon(x^*),\ j \in \underline{m}$$

$$x \in B_{x^*}$$

i.e., we can substitute a finite set of continuously differentiable inequalities for the original continuous set of inequalities.

Consider now the derivative of $\eta^j_{x,\omega}$. If $\omega \in \text{bd}\ \Omega$, we have, for ξ is some neighborhood of x,

$$\frac{\partial \tilde{\eta}^j_{x,\omega}}{\partial \xi}(\xi) = \phi^j_x(\xi, w^j_{x,\omega}(\xi)) = \phi^j_x(\xi,\omega) \tag{2.11}$$

and if $\omega \in \text{int}\ \Omega$, for ξ in some neighborhood of x ,

$$\frac{\partial \tilde{\eta}^j_{x,\omega}}{\partial \xi}(\xi) = \phi^j_x(\xi,w^j_{x,\omega}(\xi) + \phi^j_\omega(\xi,w^j_{x,\omega}(\xi))\frac{\partial w^j_{x,\omega}}{\partial \xi}(\xi)$$

$$= \phi^j_x(\xi,w^j_{x,\omega}(\xi)) \tag{2.12}$$

since $\phi^j_\omega(\xi,w^j_{x,\omega}(\xi)) = 0$ by definition of $\tilde{\Omega}^j_\varepsilon(\xi)$. Before proceeding further, we introduce a second assumption (the meaning of subscript ∞ is clear).

Assumption 2.2. For all x in $\mathbb{R}^n$, the sets $\tilde{\Omega}^j_\infty(x)$, $j \in \underline{m}$ are finite.

$\square$

We shall denote by $k^j_\varepsilon(x)$ the cardinality of $\tilde{\Omega}^j_\varepsilon(x)$. For the purpose of simplicity of notation in stating on algorithms, we define the "pseudo-functions"(*) $\eta^{ij}_\varepsilon(x)$, by

$$\eta^{ij}_\varepsilon(x) \triangleq \phi^j(x,\omega^{ij}_\varepsilon(x)), \quad i = 1,..,k^j_\varepsilon(x), \tag{2.13}$$

where $\omega^{ij}_\varepsilon(x)$ is the ith element of $\tilde{\Omega}^j_\varepsilon(x)$ considered as ordered by "$<$". We now see that, if x* satisfies (i), (ii) and (iii) and if $k^j_\varepsilon(x)$ is constant in a neighborhood of x* then, in that neighborhood,

$$\{\eta^{ij}_\varepsilon(x) \mid i = 1,..,k^j_\varepsilon(x)\} = \{\tilde{\eta}^j_{x*,\omega}(x) \mid \omega \in \tilde{\Omega}^j_\varepsilon(x*)\} \quad .$$

We will see later how we can devise algorithms exploiting this fact, by using a modification of the recursive quadratic programming technique, which was first introduced by Wilson [7] for the standard nonlinear programming problem and later analyzed by Robinson [8], see also [9,10,11,12,13,14]. However, first we need to introduce some more notations. Remembering that expressions (2.11) and (2.12) are valid only

(*) By this we want to emphasize the fact that $\eta^{ij}_\varepsilon(x)$ is defined only for all x s.t. $i \leq k^j_\varepsilon(x)$.

if x is such that (i), (ii) and (iii) are satisfied, we get around the difficulty by defining a "pseudo-gradient" $\overline{\nabla}\eta_\varepsilon^{ij}(x)$ by

$$\overline{\nabla}\eta_\varepsilon^{ij}(x) \triangleq \phi_x^j(x,\omega_\varepsilon^{ij}(x))^T, \quad i = 1,..,k_{\varepsilon_0}^j(x), \tag{2.14}$$

Similar considerations lead us to define a "pseudo-Hessian" $\overline{\nabla^2}\eta_\varepsilon^{ij}(x)$ by

$$\overline{\nabla^2}\eta_\varepsilon^{ij}(x) \triangleq \phi_{xx}^j(x,\omega^{ij}(x)), \quad i = 1,..,k_\varepsilon^j(x) \tag{2.15}$$

which is equal to the Hessian $\nabla^2\eta_\varepsilon^{ij}(x)$ whenever the latter is well-defined. Finally we introduce an appropriate Lagrangian. We define [(*)] $L_\varepsilon : \mathbb{R}^n \times (\mathbb{R}^p)^{\mathbb{N}} \times \mathbb{R}^m \to \mathbb{R}$ together with its "pseudo-gradient"and "pseudo-Hessian" as follows:

$$L_\varepsilon(x,\mu,\lambda) = f(x) + \sum_{j=1}^{p} \sum_{i=1}^{k_\varepsilon^j(x)} \mu^{ij}\eta_\varepsilon^{ij}(x) + \sum_{j\in J_\varepsilon(x)} \lambda^j g^j(x) \tag{2.16}$$

$$\overline{\nabla}L_\varepsilon(x,\mu,\lambda) = \nabla f(x) + \sum_{j=1}^{p} \sum_{i=1}^{k_\varepsilon^j(x)} \mu^{ij}\overline{\nabla}\eta_\varepsilon^{ij}(x) + \sum_{j\in J_\varepsilon(x)} \lambda^j \nabla g^j(x) \tag{2.17}$$

$$\overline{\nabla^2}L_\varepsilon(x,\mu,\lambda) = \nabla^2 f(x) + \sum_{j=1}^{p} \sum_{i=1}^{k_\varepsilon^j(x)} \mu^{ij}\overline{\nabla^2}\eta_\varepsilon^{ij}(x) + \sum_{j\in J_\varepsilon(x)} \lambda^j \nabla^2 g^j(x) . \tag{2.18}$$

3. <u>A Conceptual, Local Quadratically Convergent Algorithm</u>.

We begin our development of an algorithm by first defining a locally convergent, conceptual version.

<u>Algorithm 3.1</u>.

<u>Parameters</u>. $\varepsilon > 0$

(*) $(\mathbb{R}^p)^{\mathbb{N}}$ is the space of infinite sequences in $\mathbb{R}^p$

Data. $x_0 \in \mathbb{R}^n$, $\lambda_0 \in \mathbb{R}^m$, $\mu_0 \in (\mathbb{R}^p)^{\mathbb{N}}$, i.e. $\{\mu_0^{i,j}\}_{\substack{i\in\mathbb{N}\\ j=1,..,p}}$

Step 0. Set $\ell = 0$.

Step 1. Solve the following quadratic program[(*)]:

$$\min_v \langle \nabla f(x_\ell),v\rangle + \frac{1}{2}\langle v,\overline{\nabla^2 L}_\varepsilon(x_\ell,\mu_\ell,\lambda_\ell)v\rangle$$

subject to $\eta_\varepsilon^{ij}(x_\ell) + \langle \overline{\nabla}\eta_\varepsilon^{ij}(x_\ell),v\rangle \leq 0$ for $i=1,..,k_\varepsilon^j(x_\ell)$, $j=1,...,p$

$$g^j(x_\ell) + \langle \nabla g^j(x_\ell), v\rangle \leq 0 \text{ for } j \in J_\varepsilon(x) \quad ; \tag{3.1}$$

obtain v_ℓ and corresponding multipliers $\lambda_{\ell+1}$ and $\mu_{\ell+1}$.

Step 2. Set $x_{\ell+1} = x_\ell + v_\ell$ and $\mu_{\ell+1}^{ij} = 0$ for all $i \in [k_\varepsilon^j(x_\ell)+1, k_\varepsilon^j(x_{\ell+1})]$, $j \in \underline{p}$; set $\ell = \ell + 1$ and go to Step 1. $\square$

The above algorithm is obviously related to the method of recursive quadratic programming for the standard nonlinear programming problem. Hence we can hope that under suitable assumptions it will exhibit a local quadratic rate of convergence. Note however that, since local maxima of $\phi(x,\cdot)$ cannot be computed exactly, Algorithm 3.1 is only conceptual; in section 5, we will introduce an implementable version. Note also that, unless we are close to a local solution of (2.1) our way of updating the μ's does not seem too promising, since, when the sets $\tilde{\Omega}_\varepsilon^j(x)$ vary in cardinality, some of the μ^{ij}'s will be associated with constraints they were not meant for, since in (2.16)-(2.18) they are assigned in increasing order of the maximizers. We will come back to this question in section 4. Before proceeding further, let us introduce a few more assumptions, which formalize our stipulations (i), (ii) and

(*) throughout this paper, by "solving a quadratic program" we will mean finding its Kuhn-Tucker point of minimal norm

(iii) in section 2. Our assumptions will be in terms of a local solution for problem (2.1), i.e. a point x^* such that $x^* \in F$ and $f(x) \geq f(x^*)$ for all $x \in F \cap B(x^*,\rho)$, for some $\rho > 0$. We denote by S the set of local solutions to problem (2.1).

<u>Assumption 3.1</u>. Suppose that $x^* \in S$ and $j \in \underline{p}$

(i) If $\omega \in \tilde{\Omega}_\infty^j(x^*)$, then if $\omega \in \text{int}\,\Omega$, $\phi_{\omega\omega}^j(x^*,\omega) \neq 0$ and if $\omega \in \text{bd}\,\Omega$, $\phi_\omega^j(x^*,\omega) \neq 0$.

(ii) If $\omega \in \Omega_\infty^j(x^*)$ and $\phi_\omega^j(x^*,\omega) = 0$, then $\phi_{\omega\omega}^j(x^*,\omega) \neq 0$. ¤

<u>Assumption 3.2</u>. For any $x^* \in S$ and $\omega \in \tilde{\Omega}_\varepsilon(x^*)$,

$$\phi(x^*,\omega) + \varepsilon \neq 0 \quad . \qquad ¤$$

<u>Assumption 3.3</u>. For any $x^* \in S$,

$$\{\nabla g^j(x^*),\ j \in J_0(x^*);\ \phi_x^j(x^*,\omega)^T,\ \omega \in \tilde{\Omega}_0(x^*),\ j \in \underline{p}$$

is a set of linearly independent vectors. ¤

Note that if S is finite Assumption 3.2 should not cause any difficulty : it will be "almost surely" satisfied. Our first result is

<u>Lemma 3.1</u>.

Let Assumption 2.1, 2.2, 3.1 and 3.2 be satisfied and let $x^* \in S$. Then there exists an $\rho^* > 0$ such that, for all $x \in B(x^*,\rho^*)$, we have

$$k_\varepsilon^j(x) = k_\varepsilon^j(x^*), \forall j \in \underline{p}; \qquad (3.2)$$

moreover, for all $\omega \in \tilde{\Omega}_\varepsilon^j(x^*)$, $j \in \underline{p}$, the functions $\eta_{x^*,\omega}^j(\cdot)$, are continuously differentiable in $B(x^*,\rho^*)$ and for all $x \in B(x^*,\rho^*)$

$$\eta_\varepsilon^{ij}(x) = \eta^j_{x^*,\omega^{ij}}(x) \tag{3.3}$$

where ω^{ij} is the $\underline{i}$th element of $\tilde{\Omega}_\varepsilon^j(x^*)$ is increasing order; consequently $\eta_\varepsilon^{ij}(\cdot)$ is continuously differentiable in $B(x^*,\rho^*)$.

Proof.

Follows from the Implicit Function theorem, using Assumptions 2.1, 2.2,3.1 and 3.2. ¤

The above lemma is the key to local convergence of our Algorithm 3.1. To see this we first note the following fact.

Lemma 3.2.

Let Assumptions 2.1, 2.2, 3.1 and 3.2 be satisfied and let $x^* \in S$. Then

$$F \cap B(x^*,\varepsilon^*) = \{x | g^j(x) \leq 0,\ \forall\ j \in J_\varepsilon(x^*);\ \eta_\varepsilon^{ij}(x) \leq 0\ \forall\ i \in k_\varepsilon^j(x^*),$$
$$\forall\ j \in \underline{p}\} \cap B(x^*,\varepsilon^*) \tag{3.4}$$

when ε^* is given by Lemma 1.

Proof.

Obvious from Lemma 3.1 and the definition of $\eta_\varepsilon^{ij}(\cdot)$. ¤

From Lemma 3.1 and Lemma 3.2, it now follows that problem (2.1) is equivalent, in a neighborhood of x^* to the problem

$$\min\{f(x) | g^j(x) \leq 0,\ \forall\ j \in J_\varepsilon(x^*);\ \eta_\varepsilon^{ij}(x) \leq 0, \forall\ i \in k_\varepsilon(x^*)\}, \forall\ j \in \underline{p}\} \tag{3.5}$$

In order to prove quadratic convergence of Algorithm 3.1, we need to adapt to problem (2.1) the well-known notion of a strong local solution.

<u>Definition 3.1</u>. Let $x^* \in S$. Then x^* is a strong local solution to problem (2.1) if

(i) there exist multipliers $\lambda^* \in \mathbb{R}_+^m$, $\mu^{*j} \in \mathbb{R}_+^{k_0^j(x^*)}$, $j = 1,..,p$ such that

$$\nabla f(x^*) + \sum_{i=1}^{m} \lambda^{*i} \nabla g^i(x^*) + \sum_{j=1}^{p} \sum_{i=1}^{k_0^j(x^*)} \mu^{*ij} \overline{\nabla} \eta_0^{ij}(x^*) = 0 \; {}^{(*)} \tag{3.6}$$

$$\lambda_i^* g^i(x^*) = 0 \quad , \forall i \in \underline{m} \tag{3.7}$$

(ii) $$\{\nabla g^i(x^*) \,|\, g^i(x^*) = 0\} \cup \{\overline{\nabla}\eta_0^{ij}(x^*) \,|\, i \in \underline{k_0^j}(x^*),\ j \in \underline{p}\} \tag{3.8}$$

is a linearly independent set of vectors

(iii) $$\lambda_i^* > 0 \quad \forall i \text{ such that } g^i(x^*) = 0 \tag{3.9}$$

$$\mu^{*ij} > 0 \quad \forall i \in k_o^j(x^*),\ j \in \underline{p} \tag{3.10}$$

(iv) $$\langle y, \overline{\nabla^2} L(x^*, \mu^*, \lambda^*) y \rangle > 0 \text{ for all } y \neq 0 \text{ such that} \tag{3.11}$$

$$\langle y, \nabla g^i(x^*) \rangle = 0 \ \forall i \text{ such that } \ g^i(x^*) = 0$$

$$\langle y, \overline{\nabla}\eta_0^{ij}(x^*) \rangle = 0 \quad \forall i \in \underline{k_0^j}(x^*),\ j \in \underline{p}$$

The triplet (x^*, λ^*, μ^*) is then called a strong Kuhn-Tucker triplet where $\mu^* = (\mu^{*1}, \ldots, \mu^{*p})$. ◻

(*) with the convention that $k_0^j(x^*) = 0$ implies that $\sum_{i=1}^{k_0^j(x^*)} \mu^{ij} \overline{\nabla} \eta_0^{ij}(x^*) = 0$.

Theorem 3.1. Let $z^* = (x^*,\lambda^*,\mu^*)$ be a strong Kuhn-Tucker triplet for (2.1) and suppose that Assumptions 2.1, 2.2, and 3.1-3.3 hold. Then there exists a $\rho > 0$ such that, if $(x_o,\lambda_o,\mu_o) \in B(z^*,\rho)$, Algorithm 3.1 constructs an infinite sequence $\{x_\ell,\lambda_\ell,\mu_\ell\}$ which converges to z^* R-quadratically.

Proof.

This theorem is a direct consequence of Lemma 3.1 and 3.2 and of Theorem 3.1 in [11] and the fact that, for x in a neighborhood of x^*, pseudo gradients and Hessians are identical to gradients and Hessians, i.e.,

$$\overline{\nabla}\eta^{ij}(x) = \frac{\partial\eta^{ij}(x)}{\partial x} \tag{3.12}$$

$$\overline{\nabla^2}\eta^{ij}(x) = \frac{\partial^2\eta^{ij}(x)}{\partial x^2} \tag{3.13}$$

Assumption 3.3 insures that the quadratic program in Step 2 of Algorithm 3.1 has a solution when ρ is small enough. ¤

Computational Considerations.

There is a tradeoff involved in the choice of parameter ε. On the one hand, a small value of ε will reduce then number of gradient evaluations necessary to set up the quadratic program. On the other hand, if ε is large, the quadratic program will be more likeley to have a solution; for instance, if $\overline{\nabla^2}L_\varepsilon(x_\ell,\mu_\ell,\lambda_\ell)$ is not positive definite, constraints are indispensable for the quadratic program to be bounded. Also for ε large enough, all local maxima will always be contained in the ε-strip and the algorithm will not suffer from local maxima entering and leaving the quadratic program (incidentally, Assumption 3.2 will then be satisfied for sure). If one decides to use a large ε,

$\varepsilon = \infty$ would probably be as good a choice as any.

Computational efficiency might also be improved by the following modification. In solving the quadratic program, we might drop the constraints associated with zero multipliers, thus saving gradient computations and decreasing the size of the quadratic program. All our results will still be valid since, when (x,λ,μ) is close to z^*, zero multipliers correspond to constraints inactive at z^*, and also inactive for the quadratic program in a neighborhood of z^*. However, the tradeoff here is that, away from z^*, the quadratic program might become unbounded.

4. A Conceptual Stabilized Algorithm.

In section 3 we have presented an algorithm which solves problem (2.1) with an R-quadratic rate of convergence. However, this algorithm requires that the initial triplet (x_o,λ_o,μ_o) be close to a strong Kuhn-Tucker triplet. In this section, we will construct a mechanism for initializing Algorithm 3.1. The mechanism consists of two blocks. The first block is any globally convergent algorithm yielding points close to some local solution; we shall use the algorithm proposed by Gonzaga, Polak and Trahan [4]. The second block is any test for detecting when the sequence constructed by the globally convergent algorithm is sufficiently close to a strong local solution for the local quadratically convergent algorithm to take over; we shall use a test related to the one in Algorithm Model 3 in [15]. Let $A_1(\cdot)$ be defined by Steps 2 and 3 of Algorithm 3.1 (whenever the quadratic program or Step 2 has a solution), so that an infinite sequence $\{(x_\ell,\lambda_\ell,\mu_\ell)\} = \{z_\ell\}$ generated by Algorithm 3.1 satisfies $z_{\ell+1} = A_1(z_\ell)$, $\ell = 0,1,2,\ldots$.

Gonzaga, Polak and Trahan [4] present a first order algorithm which is a "phase I-phase II" feasible direction algorithm for problem (2.1). This algorithm constructs search directions by evaluating the optimality function $\theta_\varepsilon(x)$ defined, for $\varepsilon \geq 0$ and $x \in \mathbb{R}^n$, by

$$\theta_\varepsilon(x) = \max\{-\tfrac{1}{2}\|\mu^0 \nabla f(x) + \sum_{j \in J_\varepsilon(x)} \lambda^j \nabla g^j(x) + \sum_{j=1}^{p} \sum_{i=1}^{k_\varepsilon(x)} \mu^{ij} \overline{\nabla} \eta_\varepsilon^{ij}(x)\|^2 - \gamma \mu^o \psi(x)$$

$$+ \sum_{j \in J_\varepsilon(x)} \lambda^j (g^j(x) - \psi(x)_+) + \sum_{j=1}^{p} \sum_{i=1}^{k_\varepsilon(x)} \mu^{ij} (\eta_\varepsilon^{ij}(x) - \psi(x)_+) \mid$$

$$\mu^0 + \sum_{j \in J_\varepsilon(x)} \lambda^j + \sum_{i=1}^{k_\varepsilon(x)} \mu^i = 1,\ \mu^0 \geq 0,\ \mu^{ij} \geq 0,\ \lambda^j \geq 0\} \quad . \tag{4.1}$$

Let the solution of the quadratic program (4.1) be denoted by $\mu_\varepsilon^0(x) \in \mathbb{R}$, $\lambda_\varepsilon(x) \in \mathbb{R}^m$ and $\mu_\varepsilon(x) \in (\mathbb{R}^p)^{\mathbb{N}}$. We define the search direction $h_\varepsilon(x)$ by

$$h_\varepsilon(x) = -\{\mu_\varepsilon^0(x) \nabla f(x) + \sum_{j \in J_\varepsilon(x)} \lambda_\varepsilon^j(x) \nabla g^j(x)$$

$$+ \sum_{j=1}^{p} \sum_{i=1}^{k_\varepsilon(x)} \mu_\varepsilon^{ij}(x)\, \overline{\nabla} \eta_\varepsilon^{ij}(x)\} \quad . \tag{4.2}$$

For our problem, the algorithm proposed in [4] assumes the following form.

<u>Algorithm 4.1</u>. (Gonzaga, Polak, Trahan)

<u>Parameter</u>. $\alpha, \beta \in (0,1)$, $\delta \in (0,1]$, $\gamma \geq 1$, $\varepsilon_0 > 0$.

<u>Data</u>, $x_0 \in \mathbb{R}^n$.

<u>Step 0</u>. Set $\ell = 0$.

Step 1. Set $\varepsilon = \varepsilon_0$.

Step 2. Compute $\theta_\varepsilon(x_\ell)$ and $h_\varepsilon(x_\ell)$ via (4.1) and (4.2).

Step 3. If $\theta_\varepsilon(x_\ell) \leq -\delta\varepsilon$ go to Step 4. Else set $\varepsilon = \varepsilon/2$ and go to Step 2.

Step 4. If $\psi(x_i) \leq 0$, compute the largest step size $\sigma_\ell = \beta^{q_\ell}$

(q_ℓ nonnegative integer) such that

$$f(x_\ell+\sigma_\ell h_\varepsilon(x_\ell)) - f(x_\ell) \leq -\alpha\sigma_\ell\delta\varepsilon \tag{4.3}$$

$$\psi(x_\ell+\sigma_\ell h_\varepsilon(x_\ell)) \leq 0 \quad . \tag{4.4}$$

If $\psi(x_i) \geq 0$ compute the largest step size $\sigma_\ell = \beta^{q_\ell}$ (q_ℓ nonnegative integer) such that

$$\psi(x_\ell+\sigma_\ell h_\varepsilon(x_\ell)) - \psi(x_\ell) \leq -\alpha\sigma_\ell\delta\varepsilon \quad . \tag{4.6}$$

Step 5. Set $x_{\ell+1} = x_\ell + \sigma_\ell h_\varepsilon(x_\ell)$. If $\mu^0_\varepsilon(x_{\ell+1}) \neq 0$ set $\mu_{\ell+1} = \mu_\varepsilon(x_{\ell+1})/\mu^0_\varepsilon(x_{\ell+1})$ and $\lambda_{\ell+1} = \lambda_\varepsilon(x_{\ell+1})/\mu^0_\varepsilon(\gamma_{\ell+1})$. Else set $\mu_{\ell+1} = \mu_\varepsilon(x_{\ell+1})$ and $\lambda_{\ell+1} = \lambda_\varepsilon(x_{\ell+1})$. Set $\ell = \ell + 1$ and go to Step 1.

¤

We will be interested in triplets $z = (x,\lambda,\mu)$ where $x \in \mathbb{R}^n$ is an estimate of a local optimizer and $\lambda \in \mathbb{R}^m$, $\mu \in (\mathbb{R}^p)^{\mathbb{N}}$ estimates of corresponding Kuhn-Tucker multipliers. It is shown in [4] that if $\hat{x}$ is optimal for (2.1), then $\theta_0(\hat{x}) = 0$. Hence, we will define the set Δ of desirable points by (see (4.1))

$$\Delta = \{z = (x,\lambda,\mu) \,|\, \lambda \geq 0,\ \mu \geq 0, \psi(x) \leq 0 \text{ and}$$

$$\exists \mu^0 \geq 0 \text{ with } (\mu^0,\mu) \neq 0 \text{ s.t. } \mu^0 \nabla f(x)$$

$$+ \sum_{j \in J_0(x)} \lambda^j \nabla g^j(x) + \sum_{j=1}^{p} \sum_{i=1}^{k_0^j(x)} \mu^{ij} \overline{\nabla} \eta_0{}^{ij}(x) = 0 \} \,. \tag{4.7}$$

We will denote by μ^j an infinite sequence with elements μ^{ij}.

The convergence properties of Algorithm 4.1 are stated in [4] as follows

Lemma 4.1. If Algorithm 4.1 constructs a sequence $\{z_\ell\} = \{(x_\ell,\lambda_\ell,\mu_\ell)\}$ which is finite (due to infinite looping in Steps 2-3), then $(x_k,\lambda_0(x_k), \mu_0(x_k))$ is desirable, where x_k is the last value of x_ℓ. If $\{z_\ell\}$ is infinite, any accumulation point is desirable. ⌑

We can express the computations in one iteration of Algorithm 4.1 in terms of the maps A_2^ε and E_2^ε which are evaluated as follows (the definitions depend on $\alpha,\beta,\gamma,\delta$).

Procedure 4.1. Evaluation of $A_2^\varepsilon(\cdot)$ and $E_2^\varepsilon(\cdot)$.

Data. $z = (x,\lambda,\mu)$, $\varepsilon \geq 0$.

Step 0. Set $\bar{\varepsilon} = \varepsilon$.

Step 1. Compute $\theta_{\bar{\varepsilon}}(x)$, $h_{\bar{\varepsilon}}(x)$.

Step 2. If $\theta_{\bar{\varepsilon}}(x) \leq -\delta\bar{\varepsilon}$ go to Step 3 . Else, set $\bar{\varepsilon} = \bar{\varepsilon}/2$ and go to Step 1.

<u>Step 3</u>. If $\psi(x) \leq 0$, compute the largest step size $\sigma = \beta^t$ (t nonnegative integer) such that

$$f(x+\sigma h_{\bar{\varepsilon}}(x)) - f(x) \leq -\alpha\sigma\delta\bar{\varepsilon} \qquad (4.8)$$

$$\psi(x+\sigma h_{\bar{\varepsilon}}(x)) \leq 0 \qquad (4.9)$$

If $\psi(x) > 0$, compute the largest step size $\sigma = \beta^t$ (t nonnegative integer) such that

$$\psi(x+\sigma h_{\bar{\varepsilon}}(x)) - \psi(x) \leq -\alpha\sigma\delta\bar{\varepsilon} \quad . \qquad (4.10)$$

<u>Step 4</u>. Compute $\mu^0 = \mu^0_{\bar{\varepsilon}}(x+\sigma h_{\bar{\varepsilon}}(x))$, $\lambda = \lambda_{\bar{\varepsilon}}(x+\sigma h_{\varepsilon}(x))$, $\mu = \mu_{\bar{\varepsilon}}(x+\sigma h\ (x))$. If $\mu^0 \neq 0$, set $\lambda = \lambda/\mu_0$, $\mu = \mu/\mu_0$.

<u>Step 5</u>. Set $A_2^{\varepsilon}(z) = (x+\sigma h_{\bar{\varepsilon}}(x),\lambda,\mu)$, $E_2^{\varepsilon}(z) = \bar{\varepsilon}$.

¤

<u>Remark</u>.

Obviously if this iteration is used several times in a row, Step 4 is needed only for the last one (since Step 1 of next iteration solves the same quadratic program). ¤

We are going to present two different ways of connecting A_1 and A_2^{ε} together to produce a stabilized algorithm. The first one follows a technique used previously among others by Polak and Mayne [6]; we will see that, in order to obtain satisfying convergence theorems, we need to introduce a rather strong assumption (Assumption 4.1). The second scheme has been used previously [16] to stabilize a Newton algorithm by using a multiplier method; this second scheme allows stronger theoretical results. From a practical point of view, however, it is not clear whether one of these schemes is definitely superior to the other. In most situations, both algorithms are expected to behave identically. The following assumption will be needed for the first scheme (see Polak and Mayne [6]).

Assumption 4.1. Suppose $z \in \Delta$. Then z is a strong local Kuhn-Tucker triplet. ¤

In the algorithm stated below, A_1 depends implicitly on a parameter $\varepsilon > 0$ (possibly ∞) as described in section 2. This constant parameter is not to be confused with the variable ε_ℓ used in the stabilizing algorithm.

Algorithm 4.2a. (stabilized, version 1).

Parameters. $\alpha, \beta \in (0,1)$, $\gamma \geq 1$, $\delta \in (0,1)$, $\eta \in (0,1)$, $K > 0$, $\varepsilon_0 > 0$, $\varepsilon > 0$, $\nu \in (0,1)$.

Data. $x_0 \in \mathbb{R}^n$, $\lambda_0 \in \mathbb{R}^m$, $\mu_0 \in (\mathbb{R}^p)^{\mathbb{N}}$.

Step 0. Set $\ell = 0$, $j = 0$, $z_0 = (x_0,\mu_0)$, $\bar{\varepsilon} = \varepsilon_0$.

Step 1. If $A_1\ (z_\ell)$ is well defined and $\|A_1(z_\ell) - z_\ell\| \leq K\eta^j$ set $z_{\ell+1} = A_1(z_\ell)$, $j = j+1$, $\ell = \ell + 1$ and go to Step 1.

Step 2. Set $z_{\ell+1} = A_2^{\varepsilon_\ell}(z_\ell)$, $\varepsilon_{\ell+1} = E_2^{\varepsilon_\ell}(z_\ell)$, $\ell = \ell + 1$.

Step 3. If $\varepsilon_\ell \le \nu\bar{\varepsilon}$ go to step 2. Else proceed.

Step 4. Set $\bar{\varepsilon} = \varepsilon_\ell$. Compute $\theta_\varepsilon(x_\ell)$, to obtain $\mu^0_\varepsilon(x_\ell)$, $\mu_\varepsilon(x_\ell)$. Set $\mu^0 = \mu^0_\varepsilon(x_\ell)$, $\lambda = \lambda_\varepsilon(x_\ell)$, $\mu = \mu_\varepsilon(x_\ell)$. If $\mu^0 \neq 0$, set $\lambda = \lambda/\mu_0$, $\mu = \mu/\mu_0$. Set $z_\ell = (x_\ell,\lambda,\mu)$ and go to Step 1. ⌑

Theorem 4.1. Suppose that the sequence $\{z_\ell\}$ is constructed by Algorithm 4.2a and that Assumptions 2.1, 2.2, 3.1-3.3 and 4.1 are satisfied. Under these conditions

(i) if $\{z_\ell\}$ is finite then the last element is desirable.

(ii) if $\{z_\ell\}$ is infinite and bounded, then $z_\ell \to z^*$, R-quadratically, with z^* a strong local triplet. ⌑

We first prove two lemmas.

Lemma 4.2. Suppose that Assumptions 2.1, 2.2, and 3.1-3.3 are satisfied and that Algorithm 4.2a constructs an infinite sequence $\{z_\ell\}$ such that there exists an ℓ_0 such that for $\ell \ge \ell_0$, z_ℓ is constructed in Step 1. Then $z_\ell \to \hat{z}$ with $\hat{z} \in \Delta$.

Proof.

The proof of this lemma is identical to the first part of the proof of Lemma 3.1 in [6], bearing in mind the definition (4.7) for a desirable point. ⌑

Lemma 4.3. Suppose Algorithm 4.2a constructs an infinite sequence $\{z_\ell\}$, containing an infinite subsequence $\{z_\ell\}_K$ which is constructed in Step 2 and which satisfies the test in Step 2. Then any accumulation point of $\{z_\ell\}_K$ is desirable.

Proof.

Let $z_\ell \overset{K'}{\rightarrow} \hat{z}$ with $K' \subset K$. Since, for any $\ell_1, \ell_2 \in K'$, with $\ell_2 > \ell_1$,

$$\varepsilon_{\ell_2} < \nu\varepsilon_{\ell_1} \quad , \tag{4.15}$$

we know that $\varepsilon_\ell \to 0$ as $\ell \to \infty$, $\ell \in K'$. Also, we have

$$\varepsilon_{\ell_i} < \varepsilon_{\ell_{i-1}} \text{ for } i = 1,2 \tag{4.16}$$

which implies that

$$\varepsilon_{\ell_i} = E_2^{\varepsilon_0}(z_{\ell_i}) \text{ for } i = 1,2 \quad . \tag{4.17}$$

It has been shown (Lemma 1 in [17]) that this implies

$$\theta_0(\hat{x}) = 0 \quad . \tag{4.18}$$

Since it follows from our choice (4.1) of an optimality function that $\mu_\varepsilon^0(x)$, $\lambda_\varepsilon(x)$ and $\mu_\varepsilon(x)$ as they are constructed in Step 4 converge to $\mu_0^0(x)$, $\lambda_0(\hat{x})$ and $\mu_0(\hat{x})$, we conclude that $\hat{z} \in \Delta$. ¤

Proof of Theorem 4.1. First suppose that, for $\ell \geq \ell_0$, $\{z_\ell\}$ is entirely constructed in Step 1. By Lemma 4.2 and Assumption 4.1, $z_\ell \to z^*$, where z^* is a strong Kuhn-Tucker triplet. From Theorem 3.1, we conclude that convergence in R-quadratic. Second, suppose that there exists an infinite subsequence $\{z_\ell\}_K$ such that z_ℓ is constructed in Step 2. Then, it follows from Lemma 4.1 and from Lemma 1 in [17] that there exists a further subsequence indexed by $K' \subset K$ such that

$$\varepsilon_{\ell_2} < \nu\varepsilon_{\ell_1} \; \forall \; \ell_2 > \ell_1, \quad \ell_1, \ell_2 \in K' \tag{4.19}$$

Using Lemma 4.3 and Assumption 4.1 we conclude that $z_\ell \xrightarrow{K'} z^*$, a strong Kuhn-Tucker triplet. Again, by Theorem 3.1 we conclude that convergence is R-quadratic. ¤

We now introduce our second stabilized version.

Algorithm 4.2b (stabilized, version 2)

Parameters. $\alpha, \beta \in (0,1)$, $\gamma \geq 1$, $\delta \in (0,1]$, $\eta \in (0,1)$, $K > 0$.

Data. $x_0 \in \mathbb{R}^n$, $\lambda_0 \in \mathbb{R}^m$, $\mu_0 \in (\mathbb{R}^p)^{\mathbb{N}}$.

Step 0. Set $\ell = 0$, $j = 0$, $z_0 = (x_0, \lambda_0, \mu_0)$.

Step 1. If $A_1(z_\ell)$ is well defined and $\|A_1(z_\ell) - z_\ell\| \leq K\eta^j$ then go to Step 2. Else go to Step 3.

Step 2. If z_ℓ was constructed in Step 3 or if $\ell = 0$, then set $\tilde{\ell} = \ell$, $\tilde{j} = j$ and $\tilde{z} = z_\ell$. Set $z_{\ell+1} = A_1(z_\ell)$, $j = j + 1$, $\ell = \ell + 1$ and go to Step 1.

Step 3. If z was constructed in Step 2, then set $\ell = \tilde{\ell}$, $j = \tilde{j}$ and $z_\ell = \tilde{z}$. Set $z_{\ell+1} = A_2^{\varepsilon_\ell}(z_\ell)$, $\varepsilon_{\ell+1} = E_2^{\varepsilon_\ell}(z_\ell)$, $\ell = \ell + 1$. Compute $\theta_\varepsilon(x_\ell)$ to obtain $\mu_\varepsilon^0(x_\ell)$, $\lambda_\varepsilon(x_\ell)$, $\mu_\varepsilon(x_\ell)$. Set $\mu^0 = \mu_\varepsilon^0(x_\ell)$, $\lambda = \lambda_\varepsilon(x_\ell)$, $\mu = \mu_\varepsilon(x_\ell)$. If $\mu^0 \neq 0$, set $\lambda = \lambda/\mu_0$, $\mu = \mu/\mu_0$. Set $z_\ell = (x_\ell, \mu)$ and go to Step 1.

Theorem 4.2. Suppose that the sequence $\{z_\ell\}$ is constructed by Algorithm 4.2b and that Assumptions 2.1, 2.2 and 3.1 - 3.3 are satisfied. Then

(i) if $\{z_\ell\}$ is finite the last element is desirable.

(ii) if $\{z_\ell\}$ is infinite then any accumulation point of $\{z_\ell\}$ is desirable and if any such accumulation point z^* is a strong local triplet, then $z_\ell \to z^*$ R-quadratically.

Proof:

Clearly, the algorithm can behave in two different ways. Either $\{z_\ell\}$ is entirely constructed in Step 2 or, if there is an ℓ_0 such that z_{ℓ_0} is constructed in Step 1, then z_ℓ is constructed in Step 1 for all $\ell \geq \ell_0$. In both cases all accumulation points are desirable (by Lemma 4.1 or Lemma 4.2). Let us show that in the first case, $\{z_\ell\}$ does not have any accumulation point which is a strong local triplet. For suppose that $z_\ell \xrightarrow{K} z^*$ for some K, with z^* a strong local triplet. Then the sequence would eventually reach a z_{ℓ_0} such that, by Theorem 3.1, $\{z_\ell\}$ would be constructed in Step 1 from then on. Hence, if any accumulation point z^* is a strong local triplet, then the tail of the sequence is constructed in Step 1 and, by the same argument as in Theorem 4.1, $z_\ell \to z^*$ R-quadratically. ⌑

As mentioned earlier, it appears that Algorithm 4.2b results in a stronger convergence theorem than Algorithm 4.2a (because Assumption 4.1 is not needed) and, in addition, the proof is much simpler. However, as far as efficiency is concerned, the comparison is more delicate. Indeed, it may appear wasteful not to retain anything from the computations performed in Step 1 as soon as Step 2 is reentered. However the following two facts have to be taken into consideration. First, if

Step 2 is reentered after Step 1, then we must conclude that Step 1 was entered improperly, probably too far away from a strong solution, and hence the effect of using A_1, in that case, is unpredictable. Second, even if the effect of using A_1 is beneficial, since the stepsize is limited to a small value by the test is Step 1, the possible improvement obtained in the Step 1 iterationsthat have been performed is probably not very large.

Computational Considerations.

In section 3 it was pointed out that, away from a strong Kuhn-Tucker triplet, the multipliers provided by A_1 may be very poor estimates, as in particular they may be assigned to local maxima they were not meant for. A way to circumvent this difficulty would be to solve $\theta_\varepsilon(x_\ell)$ at each iteration to give proper estimates to the multipliers to be fed to the quadratic program. The tradeoff is that such a modification might prevent quadratic convergence.

5. An Implementable Stabilized Algorithm.

The algorithm given in section 4 is conceptual since computation of the set of local maximizers cannot be performed exactly. To make the algorithm implementable we approximate $\Omega \triangleq [\omega_0,\omega_c]$ by the finite set

$$\Omega_q \triangleq \{\omega | \omega = \omega_0 + k\Delta_q, \; k = 0,1,..,q\} \tag{5.1}$$

where $\Delta_q = (\omega_c-\omega_0)/q$ (5.2)

The points in Ω_q will be referred to as mesh points. Similarly, $\zeta_q(\cdot)$, an approximation to $\zeta^j(\cdot)$ is defined by

$$\zeta_q^j(x) = \max\{\phi^j(x,\omega) | \omega \in \Omega_q\} \quad . \tag{5.3}$$

The function $\psi_q(\cdot)$ is defined by

$$\psi_q(x) = \max\{g^j(x),\ j \in \underline{m};\ \zeta_q^j(x),\ j \in \underline{p}\} \tag{5.4}$$

The approximate ε-most-active constraint set and the corresponding local maximizing set and index set are defined as follows :

$$\Omega^j_{q\,\varepsilon}(x) \triangleq \{\omega \in \Omega_q \mid \phi^j(x,\omega) \geq \psi_q(x)_+ - \varepsilon\} \tag{5.5}$$

$$\tilde{\Omega}^j_{q\,\varepsilon})x) \triangleq \{\omega \in \Omega^j_{q,\varepsilon}(x) \mid \omega \text{ is a left local maximizer of } \phi^j(x,\cdot) \text{ restricted to } \Omega^j_{q\varepsilon}(x)\} \tag{5.6}$$

$$J_{q\,\varepsilon}(x) = \{j \in \underline{m} \mid g^j(x) \geq \psi_q(x)_+ - \varepsilon\} \ . \tag{5.7}$$

A left local maximizer is defined as follows: if

$\Omega^j_{q\varepsilon}(x) = \{\overline{\omega}_1,\overline{\omega}_2,\ldots,\overline{\omega}_s\}$ then $\overline{\omega}_i$ is a left local maximizer of $\phi^j(x,\cdot)$ in $\Omega^j_{q\,\varepsilon}(x)$ if $\phi^j(x,\overline{\omega}_i) > \phi^j(x,\overline{\omega}_{i-1})$ and $\phi^j(x,\overline{\omega}_i) \geq \phi^j(x,\overline{\omega}_{i+1})$ (with the convention that $\overline{\omega}_0 = \overline{\omega}_{s+1} = -\infty$). We denote by $k^j_{q\varepsilon}(x)$ the cardinality of $\tilde{\Omega}^j_{q\,\varepsilon}(x)$. We first discuss an implementation of $A_1(\cdot)$. We recall that $A_1(z) = \begin{Bmatrix} x + v(x) \\ \lambda(x) \\ \mu(x) \end{Bmatrix}$ where $v(x)$ and $(\lambda(x),\mu(x))$ are solution and multipliers of the quadratic program (3.1). The naive approach would be to substitute $\tilde{\Omega}^j_{q\,\varepsilon}(x)$ for $\tilde{\Omega}^j_{\varepsilon}(x)$, i.e. to consider $\eta^{ij}_{q\varepsilon}(x)$ as $\phi(x,\omega)$ for ω in $\tilde{\Omega}^j_{q\varepsilon}(x)$ and similarly for its pseudo gradient and pseudo Hessian. However it appears that with little more work we can obtain much better convergence properties. Indeed, considering the naive approach, we have to deal with the constraints

$$\phi^j(x,\omega^{ij}_{q\varepsilon} + \phi^j_x(x,\omega^{ij}_{q\varepsilon})v \leq 0, \quad \omega^{ij}_{q\varepsilon} \in \tilde{\Omega}^j_{q\varepsilon}(x) \ . \tag{5.8}$$

Heuristically, we see that, if q is large enough, although $\omega^{ij}_{q\varepsilon}$ is not in general an element of $\tilde{\Omega}_\varepsilon^j(x)$, its distance to $\tilde{\Omega}_\varepsilon^j(x)$ is of the order of the mesh Δ_q. Now suppose we constrain Δ_q to decrease as fast as $\|v\|$ where v is the solution of our quadratic program, then (5.8) will still be accurate to first order, provided that we substitute for $\phi^j(x,\omega_{q\varepsilon}^{ij})$ a first order approximation to $\phi^j(x,\omega)$, where ω is an element of $\tilde{\Omega}_\varepsilon^j(x)$ (see Mayne-Polak [5]). Heuristically we have

$$\phi^j(x,\omega) \simeq \phi^j(x,\omega_{q\varepsilon}^{ij}) + \phi_\omega^j(x,\omega_{q\varepsilon}^{ij}) \quad \delta\omega \tag{5.9}$$

where $\delta\omega$ satisfies

$$0 = \phi_\omega^j(x,\omega) = \phi_\omega^j(x,\omega_{q\varepsilon}^{ij}) + \phi_{\omega\omega}^j(x,\omega_{q\varepsilon}^{ij}) \quad \delta\omega \tag{5.10}$$

i.e.,

$$\delta\omega = -\phi_{\omega\omega}^j(x,\omega_{q\varepsilon}^{ij})^{-1}\phi_\omega^j(x,\omega_{q\varepsilon}^{ij}) \text{ (*)} \tag{5.11}$$

Expression (5.11) assumes that $\delta\omega$ is small. Hence we introduce the quantity $\delta\omega_{q\varepsilon}^{ij}(x)$ defined as follows. Let

$$d \triangleq \min(|\delta\omega|, 2\Delta_q)\text{ (**)}, \; s \triangleq \operatorname{sgn}(\delta\omega) \; . \tag{5.12}$$

If $\omega_{q\varepsilon}^{ij} = \omega_0$ and $\phi_\omega^j(x,\omega_0) < 0$, then

$$\delta\omega_{q\varepsilon}^{ij}(x) = 0 \quad . \tag{5.13}$$

If $\omega_{q\varepsilon}^{ij} = \omega_c$ and $\phi_\omega^j(x,\omega_c) > 0$, then

$$\delta\omega_{q\varepsilon}^{ij} = 0 \; . \tag{5.14}$$

(*) if ω is an endpoint, then $\delta\omega = 0$

(**) the reason for choosing $2\Delta_q$ is that, for large q and some $c > 0$, $|\delta\omega| \leq \Delta_q + c\Delta_q^2 \leq 2\Delta_q$ (see [5]).

Otherwise,

$$\text{if } \omega_0 \le \omega_{q\varepsilon}^{ij} + s\delta \le \omega_c \text{ then } \delta\omega_{q\varepsilon}^{ij}(x) = s\delta \tag{5.15}$$

$$\text{if } \omega_{q\varepsilon}^{ij} + s\delta \le \omega_0 \text{ then } \delta\omega_{q\varepsilon}^{ij}(x) = \omega_0 - \omega_{q\varepsilon}^{ij} \tag{5.16}$$

$$\text{if } \omega_c < \omega_{q\varepsilon}^{ij} + s\delta \text{ then } \delta_{q\varepsilon}^{ij}(x) = \omega_c - \omega_{q\varepsilon}^{ij}. \tag{5.17}$$

Accordingly, we now define

$$\eta_{q\varepsilon}^{ij}(x) \triangleq \phi^j(x,\omega_{q\varepsilon}^{ij}) + \phi_\omega^j(x,\omega_{q\varepsilon}^{ij})\delta\omega_{q\varepsilon}^{ij}(x) \tag{5.18}$$

$$\overline{\nabla}\eta_{q\varepsilon}^{ij}(x) \triangleq \phi_x^j(x,\omega_{q\varepsilon}^{ij}) \tag{5.19}$$

$$\overline{\nabla}^2\eta_{q\varepsilon}^{ij}(x) \triangleq \phi_{xx}^j(x,\omega_{q\varepsilon}^{ij}) \tag{5.20}$$

all three with $\omega_{q\varepsilon}^{ij}$, the $\underline{i}$th element in increasing order of $\tilde{\Omega}_{q\varepsilon}^j(x)$.

The extra work in obtaining the first order approximation on (5.18) is relatively small (two scalar function evaluations for each pair (i,j)). However, as we shall see, it results in much better convergence properties. An even better approximation would have been obtained by defining $\eta_{q\varepsilon}^{ij}(x)$, instead of (5.18), by $\phi^j(x,\omega_{q\varepsilon}^{ij}+\delta\omega_{q\varepsilon}^{ij}(x))$. However, the extra function evaluation does not provide stronger convergence properties, at least as far as theory is concerned. Also note at this point that our approximation to the pseudo Hessian has not been refined beyond expression (5.20), since this already yields a second order approximation to the cost function in the quadratic program (if Δ_q is of the same order as $\|v\|$) and the cost function in the conceptual version is already a second order approximation.

We define

$$\overline{\nabla^2}L_{q\varepsilon}(x,\mu,\lambda) = \nabla^2 f(x) + \sum_{j=1}^{p}\sum_{i=1}^{k_{q\varepsilon}^j(x)} \mu^{ij}\overline{\nabla^2}\eta_{q\varepsilon}^{ij}(x) + \sum_{j\in J_\varepsilon(x)} \lambda^j \frac{\partial^2 g^j(x)}{\partial x^2}. \tag{5.21}$$

We are now ready to state the inplementable version of Algorithm 3.1.

Algorithm 5.1.

Parameters. $\varepsilon > 0$, $\bar{c} > 0$, $\bar{q} > 0$ an integer

Data. $x_0 \in \mathbb{R}^n$, $\lambda_0 \in \mathbb{R}^m$, $\mu_0 \in (\mathbb{R}^p)^{\mathbb{N}}$

Step 0. Set $\ell = 0$, $q = \bar{q}$.

Step 1. Solve the following quadratic program

$$\min_v \langle \nabla f(x_\ell), v\rangle + \frac{1}{2}\langle v, \overline{\nabla^2 L}_{q\varepsilon}(x_\ell, \mu_\ell, \lambda_\ell)v\rangle \tag{5.22}$$

subject to

$$\eta^{ij}_{q\varepsilon}(x_\ell) + \langle \overline{\nabla}\eta^{ij}_{q\varepsilon}(x_\ell), v\rangle \leq 0,\ i = 1,\ldots,k^j_{q\varepsilon}(x_\ell),\ j = 1,\ldots,p \tag{5.23}$$

$$g^j(x_\ell) + \langle \nabla g^j(x_\ell), v\rangle \leq 0\ ,\ j \in J_\varepsilon(x)\ . \tag{5.24}$$

If the solution v is such that

$$q \leq \bar{c}/\|v\|\ , \tag{5.25}$$

set $q = q + 1$ and go to Step 1.

Step 2. Denote by v_ℓ the solution of the last quadratic program and $\lambda_{\ell+1}$, $\mu_{\ell+1}$ the corresponding multipliers.

Step 3. Set $x_{\ell+1} = x_\ell + v_\ell$ and $\mu^{ij}_{\ell+1} = 0$ for all $i \in [k^j_{q\varepsilon}(x_\ell) + 1, k^j_{q\varepsilon}(x_{\ell+1})]$, $j \in \underline{p}$, set $\ell = \ell + 1$ and go to Step 1. $\square$

Theorem 5.1. Let $z^* = (x^*,\mu^*)$ be a strong Kuhn-Tucker triplet for (2.1) and suppose Assumptions 2.1, 2.2 and 3.1-3.3 hold. Then there exists a $\rho > 0$ such that, if $(x_o,\lambda_o,\mu_o) \in B(z^*,\rho)$, Algorithm 5.1 constructs an infinite sequence $\{z_\ell\}$ which converges to z^* R-quadratically.

Proof.

Our proof is strongly inspired by the proof given by Garcia-Palomares and Mangasarian [11] for the basic recursive quadratic programming algorithm (when no functional constraint is present). Following [11], we just establish the convergence of the algorithm with an R-linear rate. We need the following lemma, a direct extension of Lemma 3.1; we state it without proof.

Lemma 5.1. Let Assumptions 2.1, 2.2, 3.1 and 3.2 be satisfied with $\varepsilon > 0$ and let $x^* \in S$. Then there exists a $\rho^* > 0$ and a $q^* > 0$ such that, for all $x \in B(x^*,\rho^*)$ and all $q \geq q^*$

$$k_{q\varepsilon}(x) = k_{\varepsilon}(x^*) \tag{5.26}$$

$\Box$

To simplify notations we assume that $m = 0$ and $p = 1$ (our results carry through easily to the general case). We denote by $Q_q(\hat{z})$ the quadratic program (generalizing (5.22)-(5.24))

$$\min_v \langle \nabla f(x), v\rangle + \frac{1}{2}\langle v, G_q(z)v\rangle$$

subject to

$$\eta^i_{q\varepsilon}(x) + \langle \nabla\eta^i_{q\varepsilon}(x), v\rangle \leq 0 \quad \forall i \in \underline{k}_{q\varepsilon}(x) \quad . \tag{5.27}$$

We introduce the function $h : \mathbb{R}^{n+k_\varepsilon(x^*)} \to \mathbb{R}^{n+k_\varepsilon(x^*)}$ and $d_q : \mathbb{R}^{n+k_{q\varepsilon}(\hat{x})} \times \mathbb{R}^{n+k_{q\varepsilon}(\hat{x})} \to \mathbb{R}^{n+k_{q\varepsilon}(\hat{x})}$ as follows

$$h(z) = \begin{bmatrix} \nabla f(x) + \sum_{i=1}^{k_\varepsilon(x^*)} \mu^i \overline{\nabla} \eta_\varepsilon^i(x) \\ \\ \mu^1 \eta_\varepsilon^1(x) \\ \vdots \\ \mu^{k_\varepsilon(x^*)} \eta_\varepsilon^{k_\varepsilon(x^*)}(x) \end{bmatrix} \tag{5.28}$$

(defining $\eta_\varepsilon^i\ (x) \triangleq 0$ for $i > k_{q\varepsilon}\ (x)$)

$$\text{and } d_q(\hat{z},z) = \begin{bmatrix} \nabla f(\hat{x}) + G_q(\hat{z})(x-\hat{x}) + \sum_{i=1}^{k_\varepsilon(x)} \mu^i \overline{\nabla} \eta_{q\varepsilon}^i(\hat{x}) \\ \\ \mu^1 (\eta_{q\varepsilon}^1(\hat{x}) + \langle \overline{\nabla} \eta_{q\varepsilon}^1(\hat{x}), x-\hat{x} \rangle) \\ \vdots \\ \mu^{k_{q\varepsilon}(\hat{x})} (\eta_{q\varepsilon}^{k_{q\varepsilon}(\hat{x})} + \langle \overline{\nabla} \eta_{q\varepsilon}^{k_{q\varepsilon}(\hat{x})}, x-\hat{x} \rangle) \end{bmatrix} . \tag{5.29}$$

The functions h and d are obtained from the equalities $h(z) = 0$ and $d_q(z,\hat{z}) = 0$ which constitute the equalities of the first order Kuhn-Tucker conditions of the original problem and of the quadratic subproblem. Extending to functional constraints a result of McCormack [18], we obtain that, whenever z^* is a strong Kuhn-Tucker triplet, $\frac{\partial h(z^*)}{\partial z}$ is nonsingular, As in [11], we define

$$\beta \triangleq \frac{3}{2} \left\| \frac{\partial h(z^*)}{\partial z}^{-1} \right\| \tag{5.30}$$

Since $h(z^*) = 0$ and $\frac{\partial h}{\partial z}\ (z^*)$ is nonsingular, there is some $\rho_1 \in (0,\rho^*)$ such that $B(z^*,\rho_1)$ contains no other zero of h, hence no other Kuhn-Tucker point. From Assumption 2.1, we conclude that there exist

$\rho \in [0,\rho_1]$, $M > 0$, $\bar{q} > 0$ such that such z^* is the unique zero of h in $\bar{B}(z^*,\frac{\rho}{2})$ and that for any z_1 and $z_2 \in B(z^*,\rho)$ and $q > \bar{q}$ we have that

(ia) $\|\frac{\partial h}{\partial z}(z_1) - \frac{\partial h}{\partial z}(z_2)\| \leq 1/2\beta$ (5.31)

(ib) $\|\frac{\partial h}{\partial z}(z_1) - d_q'(z_1,z_2)\| \leq 1/2\beta$ (5.32)

where $d_q'(z_1,z_2) = \frac{\partial}{\partial z} d(z_1,z)|_{z=z_2}$

(ii) $\|h(z_1) - d_q(z_1,z_2)\| \leq M\|z_2-z_1\|^2 + \|(\nabla^2 L(z_1) - G_q(z_1))(x_1-x_2)\|$ (5.33)

(iii) $\eta_{q\varepsilon}^i(x_1) + \bar{\nabla}\eta^i(x_1)(x_2-x_1) < 0 \quad \forall i \in \{i|\eta_\varepsilon^i(x^*) < 0\}$ (5.34)

(iv) $\mu_1^i > 0 \quad \forall i \in \{i|\mu^{*i} > 0\}$ (5.35)

The following 2 lemmas, direct extensions of lemmas due to Garcia and Mangasarian [11], are stated without proof.

Lemma 5.2. Suppose, that, with the quantities defined above, it holds $\hat{z} \in B(z^*, \frac{\rho}{2})$, $4\beta\|h(\hat{z})\| \leq \rho$ and $q \geq \bar{q}$. Then there exists unique Kuhn-Tucker triplet $\tilde{z}$ of $Q_q(\hat{z})$ in $\bar{B}(\hat{z},\frac{\rho}{2})$ and $\|\tilde{z}-\hat{z}\| \leq 2\beta\|h(\hat{z})\| \leq \rho/2$. ¤

Lemma 5.3. Let $z_0 \in B(z^*,\rho/4)$. Let $\{z_i\}$ be the sequence generated by Algorithm 5.1, with $G_q(z)$ replacing $\overline{\nabla^2 L}_{q\varepsilon}(z)$ such that, for all i, $\|G_q(z_i) - \overline{\nabla^2 L}_\varepsilon(z_i)\| \leq 1/10\beta$. Then $\{z_i\}$ exists, remains in $B(z^*,\rho/2)$ and converges R-linearly to z^*. Moreover for i = 1,2,..

$$\frac{\|z_{i+1}-z_i\|}{\|z_i-z_{i-1}\|} \leq 2\beta[M\|z_i-z_{i-1}\| + \|\nabla^2 L_\varepsilon(z_{i-1})-G_q(z_{i-1}) \frac{(x_i-x_{i-1})}{\|z_i-z_{i-1}\|}\| . \quad (5.36)$$

¤

Now let us consider $G_q(z) \triangleq \overline{\nabla^2 L}_{q\varepsilon}(z)$. From Assumption 2.1 and definition of $\overline{\nabla^2}\eta_{q\varepsilon}$ we have, for some $K_1 > 0$ and all $z \in B(z^*,\rho)$

$$\|\overline{\nabla^2}L_{q\varepsilon}(z) - \overline{\nabla^2}L_\varepsilon(z)\| \leq K_1\Delta_q \tag{5.37}$$

and, from (5.25), for some $K_2 > 0$

$$\|\overline{\nabla^2}L_{q\varepsilon}(z_i) - \overline{\nabla^2}L_\varepsilon(z_i)\| \leq K_2\|x_{i+1}-x_i\|, \ i = 1,2,\ldots \tag{5.38}$$

Hence, from (5.36), since $\|x_i-x_{i-1}\| \leq \|z_0-z_{i-1}\|$,

$$\|z_{i+1}-z_i\| \leq K\|z_i-z_{i-1}\|^2 \tag{5.39}$$

for some $K > 0$, which implies R-quadratic convergence. ¤

Our final task is to propose an implementable version of our stabilized algorithms 4.2a and 4.2b. First we denote by $A_1^q(\cdot)$ the effect of Steps 1-3 of Algorithm 4 and by $v^q(\cdot)$ the solution of the quadratic program in Step 1. Then, following [4], we make use of $\theta_{q\varepsilon}(\cdot)$ and $h_{q\varepsilon}(\cdot)$, discretized version of $\theta_\varepsilon(\cdot)$ and $h_\varepsilon(\cdot)$ obtained by replacing in (4.1) and (4.2) $J_\varepsilon(x)$, $k_\varepsilon^j(x)$, $\eta_\varepsilon^{ij}(x)$, $\psi(x)$, $\mu_\varepsilon^0(x)$, $\mu_\varepsilon(x)$ and $\lambda_\varepsilon(x)$ respectively by $J_{q\varepsilon}(x)$, $k_{q\varepsilon}^j(x)$, $\eta_{q\varepsilon}^{ij}(x)$, $\psi_q(x)$, $\mu_{q\varepsilon}^0(x)$, $\mu_{q\varepsilon}(x)$ and $\lambda_{q\varepsilon}(x)$. We now can introduce the maps $A_2^{q\varepsilon}$ and $E_2^{q\varepsilon}$, discretized versions of A_2^ε and E_2^ε, together with a new map denoted $Q_2^{q\varepsilon}$ (again the maps depend on α,β,γ and δ as well as two more scalars a, b > 0) (see [4]).

<u>Procedure 5.1</u>. Computation of $A_2^{q\varepsilon}(\cdot)$, $E_2^{q\varepsilon}(\cdot)$, $Q_2^{q\varepsilon}(\cdot)$

<u>Data</u>. $z = (x,\lambda,\mu)$, $\varepsilon > 0$, $q > 0$

<u>Step 0</u>. Set $\bar{\varepsilon} = \varepsilon$.

Step 1. Determine $\bar{q}$, smallest integer $\geq \bar{q}$ such that $\Omega^j_{q\mathcal{O}}(x)$ does not contain two adjacent mesh points (for all $j \in \underline{p}$)

Step 2. Compute $\theta_{\bar{q}\bar{\varepsilon}}(x)$ and $h_{\bar{\varepsilon}}(x)$. If

$$\theta_{\bar{q}\bar{\varepsilon}}(x) > -\delta\varepsilon,\ \varepsilon \leq a/q \text{ and } \psi_q(x)_+ \leq b/q\,, \tag{5.40}$$

set $\bar{q} = \bar{q} + 1$ and go to Step 1. If

$$\theta_{\bar{q}\bar{\varepsilon}}(x) > -\delta\varepsilon \text{ but either } \varepsilon > 2/q \text{ or } \psi_q(x)_+ > b/q\ , \tag{5.41}$$

set $\bar{\varepsilon} = \bar{\varepsilon}/2$ and go to Step 1. Otherwise proceed.

Step 3. If $\psi_{\bar{q}}(x) \leq 0$, compute the largest step size $\sigma = \beta^t$ (t non-negative integer) such that

$$f(x+\sigma h_{\bar{q}\bar{\varepsilon}}(x)) - f(x) \leq -\alpha\sigma\delta\bar{\varepsilon} \tag{5.42}$$

$$\psi_q(x+\sigma h_{\bar{q}\bar{\varepsilon}}(x)) \leq 0\ . \tag{5.43}$$

If $\psi_{\bar{q}}(x) > 0$, compute the largest step size $\sigma = \beta^t$(t nonnegative integer) such that

$$\psi_q(x+\sigma h_{\bar{q}\bar{\varepsilon}}(x) - \psi_q(x) \leq -\alpha\sigma\delta\bar{\varepsilon} \tag{5.44}$$

Step 4. Compute

$$\mu^0 = \mu^0_{\bar{q}\bar{\varepsilon}}(x+h_{\bar{q}\bar{\varepsilon}}(x)),\ \lambda = \lambda_{\bar{q}\bar{\varepsilon}}(x+\sigma h_{\bar{q}\bar{\varepsilon}}(x)),\ \mu = \mu_{\bar{q}\bar{\varepsilon}}(x+\sigma h_{\bar{q}\bar{\varepsilon}}(x)) \tag{5.45}$$

If $\mu^0 \neq 0$, set $\lambda = \lambda/\mu_0, \mu = \mu/\mu_0$.

Step 5. Set

$$A_2^{q\varepsilon}(z) = (x+\sigma h_{\bar{q}\bar{\varepsilon}}(x), \lambda, \mu) \tag{5.46}$$

$$E_2^{q\varepsilon}(z) = \bar{\varepsilon} \tag{5.47}$$

$$Q_2^{q\varepsilon}(z) = \bar{q} \tag{5.48}$$

□

We are now ready to state both versions of our implementable algorithm.

Algorithm 5.2a. (stabilized implementable, version 1)

Parameters. $\alpha,\beta \in (0,1)$, $\gamma \geq 1$, $\delta \in (0,1]$, $\eta \in (0,1)$, $\theta_0 < 0$, $K > 0$, $\nu \in (0,1)$, $\varepsilon_0 > 0$, $\varepsilon > 0$, $\bar{c} > 0$, $\bar{q} > 0$ an integer.

Data. $x_0 \in \mathbb{R}^n$, $\lambda_0 \in \mathbb{R}^m$, $\mu_0 \in (\mathbb{R}^p)^{\mathbb{N}}$

Step 0. Set $\ell = 0$, $j = 0$, $z_0 = (x_0,\lambda_0,\mu_0)$, $\bar{\varepsilon} = \varepsilon_0$, $q = \bar{q}$

Step 1. If $A_1^q(z_\ell)$ is well defined, go to Step 2, else go to Step 3.

Step 2. If $q < \bar{c}/\| v^q(z_\ell)\|$, set $q = q + 1$ and go to Step 1.
Else, if $\|v^q(z_\ell)\| \leq K\eta^j$, set $z_{\ell+1} = A_1^q(z_\ell)$, set $\ell = \ell + 1$, $j = j + 1$ and go to Step 1. Else proceed.

Step 3. Set $z_{\ell+1} = A_2^{q\varepsilon}(z_\ell)$, $\varepsilon_{\ell+1} = E_2^{q\varepsilon}(z_\ell)$, $q = Q_2^{q\varepsilon}(z_\ell)$

Step 4. If $\varepsilon_\ell \geq \nu\bar{\varepsilon}$ go to Step 3. Else proceed.

Step 5. Set $\bar{\varepsilon} = \varepsilon_\ell$. Compute $\theta_\varepsilon(x_\ell)$ to obtain $\mu_\varepsilon^0(x_\ell)$, $\lambda_\varepsilon(x_\ell)$, $\mu_\varepsilon(x_\ell)$.
Set $\mu^0 = \mu_\varepsilon^0(x_\ell)$, $\lambda = \lambda_\varepsilon(x_\ell)$, $\mu = \mu_\varepsilon(x_\ell)$. If $\mu_0 \neq 0$, set $\lambda = \lambda/\mu_0$, $\mu = \mu/\mu_0$.
Set $z_\ell = (x_\ell,\lambda,\mu)$ and go to Step 1. ◻

The following theorem is easily verified.

Theorem 5.2. Suppose that the sequence $\{z_\ell\}$ is constructed by Algorithm 5.2a and that Assumptions 2.1, 2.2, 3.1-3.3 and 4.1 are satisfied. Under these conditions

(i) if $\{z_\ell\}$ is finite then its last element is desirable,

(ii) if $\{z_\ell\}$ is infinite and bounded then $z_\ell \to z^*$ R-quadratically, with z^* a strong Kuhn-Tucker pair. $\square$

In fact, Theorem 5.2 is identical with Theorem 4.1 (dealing with the conceptual version, Algorithm 4.2a). In the same way we can state an Algorithm 5.2b, an implementable version of Algorithm 4.2b. The convergence properties of this algorithm will be identical to those of Algorithm 4.2a, as stated in Theorem 4.2.

6. Conclusion

We have obtained implementable algorithms, for solving (2.1), which converge quadratically. However, for this strong property to result in efficient computation, it is necessary to make a satisfactory choice of the values of the parameters. The choice of $\bar{c}$, in particular, is critical. The reason is that a very large value of $\bar{c}$ would result in a very large value of q, even far away from the solution. This "over-discretization" is obviously wasteful when the current estimate of the solution is poor. On the other hand if the value of $\bar{c}$ is too small, convergence will be slower due to the poor approximation of the constraints and of the Hessian of the Lagrangian. Since the stepsize of the Newton method is an estimate of the distance to a solution, a reasonable choice seems to be

$$\bar{c} = \tilde{v}\,\tilde{q}$$

where $\tilde{v}$ is the smallest stepsize one is willing to accept before stopping computation and $\tilde{q}$ is the number of meshpoints defining the required final precision.

Finally, we wish to point out that it may be possible to avoid computing second derivatives, which is generally costly. In the case of a finite number of inequality constraints, it has been shown ([11]) that if a suitable approximation is substituted for the Hessian of the Lagrangian, one preserves superlinear convergence. The condition is that the sequence of approximations converge (in a weak sense) to the exact Hessian at the solution. This result has been exploited by various authors, using either a secant method ([6]) or an update formula ([10,13]). We feel confident in predicting that a similar scheme would work in the semi-infinite case as well.

References

1. E. Polak, Algorithms for a class of computer-aided design problems: a review, Automatica, 15 (1979), pp. 531-538.
2. E. Polak, Algorithms for optimal design, University of California, Electronics Research Lab. Memo No. UCB/ERL M80/18, 1980.
3. E. Polak and D. Q. Mayne, An algorithm for optimization problems with functional inequality constraints, IEEE Transactions on Automatic Control, 21(1976), pp. 184-193.
4. C. Gonzaga, E. Polak and R. Trahan, An inproved algorithm for optimization problems with functional inequality constraints, IEEE Transactions on Automatic Control, 25(1980), pp. 49-54.
5. D. Q. Mayne and E. Polak, A quadratically convergent algorithm for solving infinite-dimensional inequalities, University of California, Electronics Research Lab. Memo No. UCB/ERL M80/11, 1980.
6. E. Polak and D. Q. Mayne, A robust secant method for optimization problems with inequality constraints, University of California, Electronics Research Lab. Memo No. UCB/ERL M79/2, 1978, also J. Optimization Theory Appl. (to appear).
7. R. B. Wilson, A Simplified Algorithm for Concave Programming, Ph.D. dissertation, Graduate School of Business Administration, Harvard University, Cambridge, Mass., 1963.
8. S. M. Robinson, Perturbed Kuhn-Tucker points and rates of convergence for a class of nonlinear programming algorithms, Mathematical Programming, 7(1974), pp. 1-16.
9. M. J. D. Powell, A fast algorithm for nonlinearly constrained optimization calculations, presented at 1977 Dunder Conference on Numerical Analysis.

10. M. J. D. Powell, The convergence of variable metric methods for nonlinearly constrained optimization calculations, in Nonlinear Programming 3, O. L. Mangasarian, R. R. Meyer and S. M. Robinson, eds., Academic Press, 1978, pp. 27-63.

11. U. M. Garcia Palomares and O. L. Mangasarian, Superlinearly convergent quasi-Newton algorithms for nonlinearly constrained optimization problems, Mathematical Programming, 11(1976), pp. 1-13.

12. S. P. Han, A globally convergent method for nonlinear programming, J. Optimization Theory Appl., 22(1977), pp. 297-309.

13. S. P. Han, Superlinearly convergent variable metric algorithms for general nonlinear programming problems, Mathematical Programming 11(1976), pp. 263-282.

14. S. P. Han, A hybrid method for nonlinear programming, in Nonlinear Programming 3, O. L. Mangasarian, R. R. Meyer and S. M. Robinson, eds., Academic Press, 1978, pp. 65-95.

15. E. Polak, On the global stabilization of locally convergent algorithms, Automatica, 12(1976), pp. 337-342.

16. A. Tits, Lagrangian Based Superlinearly Convergent Algorithms for Ordinary and Semi-Infinite Optimization Problems, Ph. D. dissertation, Department of Electrical Engineering and Computer Science, University of California, Berkeley, 1980.

17. C. Gonzaga and E. Polak, On constraint dropping schemes and optimality functions for a class of outer approximation algorithms, SIAM Journal on Control and Optimization, 17(1979), pp. 477-493.

18. G. P. McCormack, Penalty function versus nonpenalty function methods for constrained nonlinear programming problems, Mathematical Programming, 1(1971), pp. 217-238.