SIAM J. OPTIMIZATION
Vol. 4, No. 1, pp. 1–43, February 1994
© 1994 Society for Industrial and Applied Mathematics

ON SENSITIVITY ANALYSIS OF NONLINEAR PROGRAMS IN BANACH SPACES: THE APPROACH VIA COMPOSITE UNCONSTRAINED OPTIMIZATION*

ALEXANDER IOFFE†

This paper is dedicated to the memory of Alan Howard.

Abstract. This article studies local sensitivity analysis of nonlinear programming problems in Banach spaces with arbitrary sets of solutions and no a priori regularity properties. This is done by first developing a corresponding theory for unconstrained optimization involving simple composite functions and then applying this theory to general nonlinear programs by way of certain reduction principles.

Key words. nonlinear programming, composite optimization, sensitivity analysis, value function, approximate solution, variational derivative, reduction to unconstrained minimization

AMS subject classifications. 49K40, 90C31, 90C48

1. Introduction.

It has long been understood in optimization that a trade-off between constraints and nonsmoothness is often possible. The idea, especially simple when extended-real-valued functions are allowed, lies at the heart of Rockafellar's theory of convex optimization [59]. Developments in nonsmooth analysis gave a green light to its applications in nonconvex optimization. We may refer to [10], [23], [24], [33], [37], [47], and [62], among others, to see how reduction to unconstrained nonsmooth optimization can be effectively used to study existence, conditions for an extremum, and numerical procedures in various classes of constrained optimization problems. This article demonstrates that, based on this approach, a viable perturbation theory of nonlinear programming can also be developed.

We are interested in the analysis of sensitivity of the value function and the solution set to small variations of a certain "perturbation" parameter near some nominal value, say equal to zero, specifically in calculating or estimating their directional derivatives of various sorts with respect to the perturbation. These and related questions were being actively studied for many years (see, e.g., [3]–[7], [11], [13]–[18], [20]–[23], [25]–[29], [45], [46], [48], [52], [55], [60], [66]–[69], and references therein), especially in connection with finite-dimensional problems satisfying certain "nondegeneracy" assumptions, among which the following three are almost always present:

(a) the assumption that the unperturbed problem has a unique solution or a finite set of isolated solutions;

(b) constraint qualifications (regularity conditions);

(c) a second-order sufficient condition.

These conditions are of a generic nature, and one may rightfully contend that typically they are valid, especially for individual problems. However, in parametric families one may (also typically) expect that degeneracies appear for certain values of parameters and, on the other hand, there are important classes of problems in which some of the conditions are often not satisfied (as in the case of (a) for many optimization problems involving eigenvalues).

The theory developed in this article does not rely on conditions of the first two types, nor, to a certain extent, on those of the third. Along with some new formulae for sensitivity

* Received by the editors August 26, 1991; accepted for publication (in revised form) August 26, 1992. This research was supported in part by the Fund for Promotion of Science at Technion under grant 100-820.

† Department of Mathematics, The Technion, Israel Institute of Technology, Haifa 32000, Israel (MAR5100@technion.bitnet).

estimates, it implies a certain revision of our perception of the role these assumptions play in sensitivity analysis. In particular (cf. [22]),

—no constraint qualification is actually needed to obtain sensitivity estimates for the value function or to estimate the rate of convergence and compute the derivative sets of solutions. In the first case the role of constraint qualification reduces to ensuring that the standard Dini directional derivatives of the value function coincide with directional derivatives of a different kind that naturally appear in the framework of the approach via unconstrained composite optimization (Theorem 4.16). In the second case constraint qualifications are replaced by much weaker "feasibility" assumptions (Theorems 4.10–4.13).

—sensitivity analysis needs not so much sufficient conditions as such, but rather closely connected and somewhat weaker *growth conditions* subsequently postulated in this article. (We refer to [7] for a discussion of the interrelationship between the conditions.) An advantage of using the growth conditions is that parametric linear programming is included in the theory. It turns out that growth conditions (hence sufficient conditions) determine rates of convergence of (approximate) solutions as perturbation goes to zero, but play no role in calculation of the derivative of the solution map which respect to perturbation.

—assumption (a) is the one that usually guarantees the existence of directional derivatives of the value function (Theorem 4.14) in the presence of growth conditions.

Central in our approach is a new method of reduction to unconstrained minimization. To describe briefly the main idea of the method, let us consider a simple problem with only inequality constraints:

$$\begin{array}{ll} \text{minimize} & f_o(x) \\ \text{subject to} & f_i(x) \le 0, \qquad i = 1, \dots, m. \end{array} \tag{1.1}$$

Let θ be the minimal value of the cost function in the problem. Then $\bar{x}$ *is a solution to the problem if and only if* $\bar{x}$ *minimizes the function*

$$x \to f(x, \theta) = \max\{f_o(x) - \theta, f_1(x), \dots, f_m(x)\}. \tag{1.2}$$

In practice, of course, we may not know θ a priori. But there is a simple formula that allows θ to be determined by means of the function (1.1), namely,

$$\theta = \min\{\mu : \inf_x f(x, \mu) \le 0\}. \tag{1.3}$$

Thus, the constrained problem (1.1) reduces to a family of unconstrained problems (1.2). The price we pay for the reduction is an increase of the dimension of the parameter by one.

We observe further that this method of reduction does not rely on any a priori regularity assumption, and this is the basic distinction from the known reduction techniques using either "infinite penalty" as in [62] or "exact penalties" as in [19], [32], [33], and [50]. (To avoid confusion, we have to emphasize here that we talk only of unconstrained reduction schemes not depending on a posteriori knowledge of Lagrange multiplier, critical cones, etc., as in the case of augmented or generalized Lagrangians.)

In either case, however, we obtain a family of composite functions, i.e., compositions of smooth maps and convex nonsmooth functions. Section 3 is devoted to sensitivity analysis of unconstrained composite optimization for a class of composite functions, including those that appear as the result of reduction. Thus, this article should also be considered along the lines of ongoing studies of composite functions, which have been arousing increased interest of late (see [8], [9], [12], [36], [51]–[53], and [59]–[62]).

And, in conclusion, we note that, although all the principal results in this article are proved for problems in arbitrary Banach spaces (keeping in mind future applications to optimal control), no additional assumptions have been imposed that would result in extra restrictions in cases when all of the spaces are finite dimensional.

Notation. In what follows, X, Y, Z, and U are Banach spaces and D is an open set in X. All functions and maps are assumed to be *twice continuously differentiable* at every point of interest unless explicitly stated otherwise. For any function or map ϕ we denote by ϕ' and ϕ'' its first and second derivatives. If ϕ depends on more than one variable, say, $\phi = \phi(x, u)$, then we write ϕ_x and ϕ_u for the partial derivatives, so that, in particular,

$$\phi'(x,u)(h,v) = \phi_x(x,u)h + \phi_u(x,u)v,$$
$$\phi''(x,u)((h,v),(h,v)) = \phi_{xx}(x,u)(h,h) + 2\phi_{xu}(x,u)(h,v) + \phi_{uu}(x,u)(v,v).$$

We study two series of "perturbed" optimization problems. The first is the *problem of mathematical programming:*

$$(\mathbf{P}_\varepsilon(u)) \qquad \begin{array}{ll} \text{minimize} & f_o(x,u) \\ \text{subject to} & x \in D; \quad f_i(x,u) \le \varepsilon, \quad i = 1,\dots,m; \quad \|F(x)\| \le \varepsilon \end{array}$$

(ε being a nonnegative scalar) with $f_i\colon X \times U \to R, F\colon X \times U \to Y$, and u and ε being the perturbation parameters. We denote by $v_\varepsilon(u)$ the *value* function of $(\mathbf{P}_\varepsilon(u))$ (i.e., the infimum of the cost function on the constrained set) and by $S_{\varepsilon,\delta}(u) = \{x\colon x$ is admissible in $(\mathbf{P}_\varepsilon(u))$ and $f_o(x,u) \le v_\varepsilon(u) + \delta\}$ the set of *δ-solutions* of the problem.

For simplicity we also denote

$$\begin{aligned} &\mathbf{P}_o(u) = \mathbf{P}(u), && \mathbf{P}(0) = \mathbf{P}; \\ &f_i(x,0) = f_i(x), && F(x,0) = F(x); \\ &v_o(u) = v(u), && v(0) = \theta; \\ &S_{\varepsilon,o}(u) = S_\varepsilon(u,\varepsilon), && S_o(u) = S(u), S(0) = S. \end{aligned}$$

We shall always assume that

$$S \ne \emptyset \quad \text{and closed}, \quad |\theta| < \infty.$$

The second series comprises unconstrained minimization problems

$$(\mathbf{C}(u)) \qquad \text{minimize } f(x,u) = g(\Phi(x,u)) \quad \text{over } x \in D,$$

where $\Phi\colon X \times U \to Z$ and g is a convex continuous (nonsmooth) function. We denote by

$$p(u) = \inf_{x \in B} f(x,u) \quad \text{and} \quad M_\delta(u) = \{x \in D\colon f(x,u) \le p(u) + \delta\}$$

the value function and the set of δ-minima of $f(\cdot, u)$, and we also simplify the notation in specific cases as follows:

$$\mathbf{C}(0) = \mathbf{C}; \quad p(0) = \vartheta; \quad M_o(u) = M(u), \quad M(0) = M.$$

We always assume that $M \ne \emptyset$.

In the course of this article we introduce a number of technical conditions, as follows:

(PP), pseudopolyhedrality condition,

(CC), compatibility condition,

(QG), quadratic growth condition,
(UL), uniform Lipschitzness condition,
(SQG), strong quadratic growth condition for composite optimization,
(T), tameness condition for composite optimization,
(LG), linear growth condition,
(N), normality condition,
(SN), strong normality condition,
(SQG), strong quadratic growth condition for constrained optimization,
(T), tameness condition for constrained optimization.

2. Reduction theorems. The composite function that we associate with $(\mathbf{P}_\varepsilon(u))$ is

$$f(x,\mu,u)=\max\{f_o(x,u)-\theta-\mu,\max_{1\le i\le m}f_i(x,u),\|F(x,u)\|\},\tag{2.1}$$

in other words, $f=g\circ\Phi$, where

$$\Phi(x,\mu,u)=(f_o(x,u)-\theta-\mu,f_1(x,u),\dots,f_m(x,u),F(x,u)):$$
$$X\times R\times U\to Z=R^{n+1}\times Y,$$
$$g(\alpha_o,\dots,\alpha_m,y)=\max\{\alpha_o,\dots,\alpha_m,\|y\|\}.$$

Perturbations in the problem are pairs (μ,u) and we shall use for the problem itself, its value function, and solution sets the same notation: $\mathbf{C}(\mu,u),p(\mu,u)$, and $M_\delta(\mu,u)$, etc. (the presence of μ in the notation indicates that we deal with this specific version of the composite optimization problem).

If an equality constraint is present in (**P**), then

$$p(\mu,u)\ge 0\quad\forall\,\mu,u.\tag{2.2}$$

If it is absent, then adding a trivial constraint with $F(x,u)\equiv 0$, we again ensure (2.2). So, we always assume that (2.2) holds.

It is obvious, furthermore, that

$$p(0,0)=0\quad\text{and}\quad|f(x,\mu,u)-f(x,\mu',u)|\le|\mu'-\mu|,$$

so that $p(\cdot,u)$ is everywhere finite and Lipschitz with constant one.

We denote by $p'_\pm(0;(\mu,u))$ the upper and lower *Dini directional derivatives* of $p(\cdot)$ at 0 along (μ,u):

$$p'_\pm(0;(\mu,u))=\limsup_{t\searrow 0}(\inf)\,t^{-1}p(t\mu,tu);$$

if $|p'_+(0;(\mu,u))|<\infty$ (respectively, $|p'_-(0;(\mu,u))|<\infty$), we set

$$p''_+(0;(\mu,u);(\xi,w))=\limsup_{t\to 0}t^{-2}[p(t\mu+t^2\xi,tu+t^2w)-\theta-tp'_+(0;(\mu,u))]$$

and, respectively,

$$p''_-(0;(\mu,u);(\xi,w))=\liminf_{t\to 0}t^{-2}[p(t\mu+t^2\xi,tu+t^2w)-\theta-tp'_-(0;(\mu,u))].$$

These become the usual upper and lower "parabolic" second-order derivatives if the first-order upper and lower Dini derivatives coincide.

The *variational derivatives* of the first and second order to measure local behavior of the value function of $(\mathbf{P}(u))$ are defined as follows:

$$d_{\pm}v(0;u) = \lim_{\varepsilon\to 0} \limsup_{t\searrow 0} (\inf)\, t^{-1}(v_{t\varepsilon}(tu) - \theta);$$

if $|d_+v(0;u)| < \infty$ (respectively, $|d_-v(0;u)| < \infty$), then

$$d_+^{(2)}v(0;u;w) = \lim_{\varepsilon\to 0} \limsup_{t\searrow 0} t^{-2}(v_{\varepsilon(t)}(tu + t^2w) - \theta - td_+v(0;u)),$$

with $\varepsilon(t) = t^2\varepsilon$ and, respectively,

$$d_-^{(2)}v(0;u;w) = \lim_{\varepsilon\to 0} \liminf_{t\searrow 0} t^{-2}(v_{\varepsilon(t)}(tu + t^2w) - \theta - td_-v(0;u)).$$

The theorems to follow establish connection between the solution sets, value function, and its directional derivatives of $(\mathbf{C}(\mu,u))$, on one hand, and the solution sets, value function, and its variational derivative of $(\mathbf{P}_\varepsilon(u))$, on the other.

THEOREM 2.1. *For any* $\varepsilon \geq 0$

$$\inf\{\mu\colon p(\mu-\varepsilon,u)\leq\varepsilon\} \leq v_\varepsilon(u) - \theta \leq \inf\{\mu\colon p(\mu-\varepsilon,u) < \varepsilon\}. \tag{2.3}$$

Furthermore,

$$d_{\pm}v(0;u) = \inf\{\mu\colon p'_{\pm}(0;(\mu,u)) \leq 0\}, \tag{2.4}$$

so that if

$$|d_+v(0;u)| < \infty \text{ (respectively, } |d_-v(0;u)| < \infty), \tag{2.5}$$

then $p'_+(0;(\cdot,u))$ *(respectively,* $p'_-(0;(\cdot,u))$*) is everywhere finite and Lipschitz (with constant* 1*). Finally, if* (2.5) *holds, then*

$$d_+^{(2)}v(0;u;w) \leq \inf\{\eta\colon p''_+(0;(d_+v(0;u),u);(\eta,w)) \leq 0\}, \tag{2.6$_+$}$$

and, respectively,

$$d_-^{(2)}v(0;u;w) \geq \inf\{\eta\colon p''_-(0;(d_-v(0;u),u);(\eta,w)) \leq 0\}. \tag{2.6$_-$}$$

Moreover, if $d_+v(0;u)$ *and* $d_-v(0;u)$ *are finite and equal, then* (2.6$_\pm$) *become equalities.*

In the statement we adopt the standard convention

$$\inf\emptyset = \infty, \qquad \sup\emptyset = -\infty.$$

Proof. If $p(\mu-\varepsilon,u) < \varepsilon$, then for a certain x we have $\|F(x)\| < \varepsilon$, $f_i(x,u) < \varepsilon$, and $f_o(x,u) - \theta - \mu + \varepsilon < \varepsilon$, whence $v_\varepsilon(u) < \theta + \mu$. Conversely, if $v_\varepsilon(u) - \theta \leq \mu$, then for any $\delta > 0$ there is an x admissible for $(\mathbf{P}_\varepsilon(u))$ and such that $f_o(x,u) \leq \theta + \mu + \delta$. Therefore, $f(x,\mu-\varepsilon,u) \leq \varepsilon + \delta$ and $p(\mu-\varepsilon,u) \leq \varepsilon + \delta$. As this is true for any positive δ, (2.3) follows.

If $p'_+(0;(\mu,u)) \leq 0$, then $p(t\mu,tu) \leq o(t)$, and for any t there is an $x = x(t)$ such that $\|F(x,tu)\| \leq o(t)$, $f_i(x,tu) \leq o(t)$, $i = 1,\ldots,m$, and $f_o(x,tu) \leq \theta + t\mu + o(t)$. It follows that for any $\varepsilon > 0$, $v_{t\varepsilon}(tu) - \theta \leq t\mu + o(t)$, hence $d_+v(0;u) \leq \mu$. On the other hand, if $d_+v(0;u) > \mu$, then there are a $\sigma > 0$ and an $\varepsilon > 0$ such that for any $\delta > 0$ we can find a $t \in (0,\delta)$ such that $v_{t\varepsilon}(tu) > t(\mu+\sigma)$. This, in turn, implies that for any x

that is admissible in $(\mathbf{P}_{t\varepsilon}(tu))$, the inequality $f_o(x,tu)-\theta-t\mu\geq t\sigma$ holds true. Thus, $p(t\mu,tu)\geq t\cdot\min\{\sigma,\varepsilon\}$ and, consequently, $p'_+(0;(\mu,u))>0$. This proves (2.4) for the upper derivative. The proof for the lower derivative is similar.

Because $p(\cdot,u)$ is everywhere finite and Lipschitz with constant 1, we conclude that $p'_\pm(0;(\cdot,u))$ is either identically equal to plus or minus infinity or everywhere finite and Lipschitz with constant 1. This completes the proof of the second part of the theorem.

Assume that $|d_+v(0;u)|<\infty$. Then (as $p'_+(0;(\cdot,u))$ is continuous)

$$p'_+(0;(d_+v(0,u)))=0.$$

Assume further that η is such that

$$p''_+(0;(d_+v(0;u),u);(\eta,w))\leq 0.$$

This means that for any $\varepsilon>0$

$$p(td_+v(0;u)+t^2\eta,tu+t^2w)-\theta\leq\varepsilon t^2$$

if t is sufficiently small, which, in view of (2.3) and the obvious fact that $p(\cdot)$ and all its derivatives are nonincreasing functions of μ, implies that

$$v_{\varepsilon(t)}(tu+t^2w)-\theta\leq td_+v(0;u)+t^2\eta.$$

This proves (2.6_+). Likewise, assuming that $|d_-v(0;u)|<\infty$, we conclude that

$$p'_-(0;(d_+v(0,u)))=0. \tag{2.7}$$

Now if $\eta>d^{(2)}_-v(0;u;w)$, then for any $\delta>0$ and $\varepsilon>0$ there is a $t\in(0,\delta)$ such that

$$td_-v(0;u)+t^2(\eta+\varepsilon)>v_{\varepsilon(t)}(tu+t^2w)-\theta.$$

It follows from the left part of (2.3) that

$$p(td_-v(0;u)+t^2(\eta+\varepsilon),tu+t^2w)\leq 0$$

and, consequently (owing to (2.7)),

$$p''_-(0;(d_-v(0;u),u);(\eta,w))\leq 0.$$

This proves (2.6_-). The proof of the last statement is similar to that for (2.4). □

Given is a family $N_\xi(u)$ of set-valued maps from U into X, depending on one or more nonnegative parameters that we jointly denote by ξ. In what follows this will be either M_ε or $S_{\varepsilon,\delta}$. We set

$$D^{1/2}_+N(x,u)=\bigcap_{\xi>0}\limsup_{t\searrow 0}t^{-1/2}(N_{t\xi}(tu)-x),$$

$$D_+N(x,u,w)=\bigcap_{\xi>0}\limsup_{t\searrow 0}t^{-1}(N_{\xi(t)}(tu+t^2w)-x),$$

where $\xi(t)=t^2\xi$ and lim sup stands for the upper limit in the Hausdorff–Kuratowski sense.

THEOREM 2.2. *For any $\varepsilon\geq 0,\delta\geq 0$ and any u,*

$$S_{\varepsilon,\delta}(u)\subset M_{\varepsilon+\delta}(v_\varepsilon(u)-\theta,u).$$

In particular, $M = S$.

Furthermore, for any $x \in S$

$$D_+^{1/2} S(x,u) \subset D_+^{1/2} M(x,(\mu,u)) \tag{2.8}$$

for any finite $\mu \geq d_+ v(0;u)$, *and*

$$D_+ S(x,u,w) \subset D_+ M(x,(d_+ v(0;u),u),(\eta,w)) \tag{2.9}$$

for any finite $\eta \geq d_+^{(2)} v(0;u,w)$, *provided that* $d_+ v(0;u)$ *is finite.*

Proof. As

$$p(v_\varepsilon(u) - \theta, u) \geq 0,$$

we have for $x \in S_{\varepsilon,\delta}(u)$ (which is admissible in $(\mathbf{P}_\varepsilon(u))$), and satisfying $f_o(x,u) \leq v_\varepsilon(u) + \delta$, that

$$f(x, v_\varepsilon(u) - \theta, u) \leq \varepsilon + \delta \leq p(v_\varepsilon(u) - \theta, u) + \varepsilon + \delta,$$

which means that $x \in M_{\varepsilon+\delta}(v_\varepsilon(u) - \theta, u)$.

For $\varepsilon = \delta = 0, u = 0$, this implies that $S \subset M$. The converse is obvious from the definition of θ and f.

Suppose now that there is an h such that for any $\delta > 0, \varepsilon > 0$ there is an $h(t) \to h$ such that $x(t) = x + \sqrt{t}h(t) \in S_{t\varepsilon,t\delta}(tu)$ at least on a sequence of t converging to zero. For such t we have

$$f_i(x(t), tu) \leq t\varepsilon, \quad i = 1, \ldots, m, \quad \|F(x(t), tu)\| \leq t\varepsilon;$$
$$f_o(x(t), tu) \leq v_{t\varepsilon}(tu) + t\delta \leq \theta + t\mu + t\delta + o(t)$$

if $\mu \geq d_+ v(0;u)$ and, consequently,

$$f(x(t), t\mu, tu) \leq t(\varepsilon + \delta),$$

which means that $x(t) \in M_{t(\varepsilon+\delta)}(t\mu, tu)$. This proves the second inclusion. The proof of the third is similar. □

3. Composite optimization. In this section we study sensitivity analysis for the composite optimization problem $(\mathbf{C}(u))$ under an additional assumption on $g(\cdot)$ that captures certain important properties of the function $f(x,\mu,u)$ considered in §2. A general case with an arbitrary convex function $g(z)$ will be considered elsewhere.

Namely, we say that $g(\cdot)$ is *pseudopolyhedral* at z if for any $v \in Z$

$$g(z + t(v + \xi)) = g(z) + tg'(z;v) + r(z,t,\xi),$$

where $t^{-2} r(z,t,\xi) \to 0$ as $\xi \to 0$, $t \to 0$. (Here and below, $g'(z;v)$ stands for the *directional derivative* of $g(\cdot)$ at z along v.) If this convergence is uniform for z belonging to a certain set R, we say that $g(\cdot)$ is pseudopolyhedral *uniformly on* R. The following is the additional assumption on $g(\cdot)$ to be adopted throughout the section:

$$g \text{ is pseudopolyhedral uniformly on } \Phi(M). \tag{PP}$$

A polyhedral function is obviously uniformly pseudopolyhedral; also, if for all v of a neighborhood of zero we have

$$g(z + v) = g(z) + \max_{z^* \in \partial g(z)} \langle z^*, v \rangle,$$

then g is pseudopolyhedral at z. It can be easily verified that the last property is valid for the $g(\cdot)$ of the preceding section at every point of M.

We set

$$\mathcal{L}(z^*, x, u) = \langle z^*, \Phi(x, u)\rangle \quad \text{(and often write } \mathcal{L}(z^*, x) \text{ instead of } \mathcal{L}(z^*, x, 0));$$
$$\Omega(x) = \{z^* : z^* \in \partial g(\Phi(x)), \mathcal{L}_x(z^*, x, 0) = 0\};$$
$$\Omega(x, h) = \{z^* \in \Omega(x) : \mathcal{L}_{xx}(z^*, x, 0)(h, h) \geq 0\};$$
$$K(x) = \{h : \mathcal{L}_x(z^*, x, 0)h \leq 0, \forall\, z^* \in \partial g(y)\} = \{h : g'(\Phi(x); \Phi_x(x, 0)h) \leq 0\};$$
$$K_\varepsilon(x) = \{h \in X : g'(\Phi(x); \Phi_x(x, 0)h) \leq \varepsilon\|h\|\}.$$

It is natural to call $\mathcal{L}$ the *Lagrangian* of $f(\cdot)$, $\Omega(x)$ the set of *Lagrange multipliers* of f at x, and $K(x)$ the *critical cone* of f at x.

If $x \in M$, then

$$\Omega(x) \neq \emptyset \quad \text{and} \quad \max_{z^* \in \Omega(x)} \mathcal{L}_{xx}(z^*, x, 0)(h, h) \geq 0 \quad \text{for all } h \in K(x),$$

which are the well-known necessary conditions for a minimum.

We also use the following notation:

$$q(x, u) = \max_{z^* \in \Omega(x)} \mathcal{L}_u(z^*, x, 0)u;$$
$$q(u) = \inf_{x \in M} q(x, u);$$
$$\Omega_u(x) = \{z^* \in \Omega(x) : \mathcal{L}_u(z^*, x, 0)u = q(x, u)\};$$
$$Q(u) = \{x \in M : q(x, u) = q(u)\};$$
$$K_u(x) = \{h : \mathcal{L}_x(z^*, x, 0)h + \mathcal{L}_u(z^*, x, 0)u \leq q(x, u), \forall\, z^* \in \partial g(\Phi(x))\}$$
$$= \{h : g'(\Phi(x); \Phi'(x, 0)(h, u)) \leq q(x, u)\}.$$
$$Q_\varepsilon(u) = \{x \in M : q(x, u) \leq q(u) + \varepsilon\};$$
$$K_{\varepsilon u}(x) = \{h : g'(\Phi(x); \Phi'(x, 0)(h, u)) \leq q(x, u) + \varepsilon\}.$$

Finally, we denote the distance from x to a set A by $\|A - x\|$:

$$\|A - x\| = \inf\{\|y - x\| : y \in A\}.$$

Remark. It might have been more appropriate to write $\varepsilon(\|h\| + \|u\|)$ in the right-hand part of the inequality in the definition of $K_{\varepsilon u}(x)$ to ensure the complete agreement between the definitions of K_ε and $K_{\varepsilon u}$. But the potential disagreement that may occur only for $u = 0$ will not lead to any confusion in the future but will provide for considerable notational simplification.

Unconditional estimates. The basic lemma. Let $A : X \to Z$ be a linear bounded operator and $v, w \in Z$. Let $\Lambda \subset Z^*$ be a convex weak* compact set. We set

$$\Omega = \{z^* \in \Lambda : A^* z^* = 0\}$$

and define q_v, K_v, Ω_v using the same formulae as for $q(x, u), K_u(x)$, and $\Omega_u(x)$ above in which $\mathcal{L}'(z^*, x)$ and $\mathcal{L}_u(z^*, x, 0)u$ are replaced, respectively, by A and v:

$$q_v = \max\{\langle z^*, v\rangle : z^* \in \Omega\}; \qquad \Omega_v = \{z^* \in \Omega : \langle z^*, v\rangle = q_v\};$$
$$K_v = \{h \in X : \langle z^*, Ah + v\rangle \leq q_v, \forall\, z^* \in \Lambda\}.$$

LEMMA 3.1. *If $\Omega \neq \emptyset$, then for any $v, y \in Z$ the limit*

$$a = \lim_{\varepsilon \searrow 0} \lim_{t \searrow 0} \inf_{\|h'-h\| \leq \varepsilon} t^{-1}[\max_{z^* \in \Lambda} \langle z^*, Ah' + v + ty\rangle - q_v]$$

exists and is equal to

$$\max_{z^* \in \Omega_v} \langle z^*, y\rangle$$

if $h \in K_v$ and $+\infty$ otherwise.

In particular, if $h \in K_v$, then there is a curve $h(t)$ converging to h as $t \to 0$ such that

$$t^{-1}[\max_{z^* \in \Lambda} \langle z^*, Ah(t) + v + ty\rangle - q_v] \to a.$$

Proof. To prove the existence of the limit, it is enough to show that the quantities a^- and a^+ obtained by replacing $\lim_{t \searrow 0}$ by $\liminf_{t \searrow 0}$ and $\limsup_{t \searrow 0}$ are equal.

If $h \notin K_v$, that is, if $\langle z^*, Ah + v\rangle > q_v$ for some $z^* \in \Omega$, then, obviously, $a^- = \infty$. In any case

$$a^- \geq \liminf_{t \searrow 0} t^{-1}[\max_{z^* \in \Omega} \langle z^*, v + ty\rangle - q_v] \geq \max_{z^* \in \Omega_v} \langle z^*, y\rangle = \alpha > -\infty. \tag{3.1}$$

Now if $h \in K_v(x)$, then setting $z = (h' - h)/t$, we have

$$\begin{aligned} \alpha \leq a^+ &= \lim_{\varepsilon \to 0} \limsup_{t \searrow 0} \inf_{\|z\| \leq \varepsilon/t} t^{-1} \max_{z^* \in \Lambda} [t\langle z^*, Az + y\rangle + \langle z^*, Ah + v\rangle - q_v] \\ &\leq \inf_z \limsup_{t \searrow 0} \max_{z^* \in \Lambda} [\langle z^*, Az + y\rangle + t^{-1}(\langle z^*, Ah + v\rangle - q_v)]. \end{aligned} \tag{3.2}$$

Fix a z. Then for any $t > 0$ there is a $z^*(t)$ at which the inner maximum is attained. Let z^* be any weak* limit point of $z^*(t)$ as $t \to 0$. Then it follows from (3.1) and (3.2) that

$$-\infty < \alpha - \langle z^*, Az + y\rangle \leq \liminf_{t \to 0} t^{-1}(\langle z^*(t), Ah + v\rangle - q_v),$$

which (as $h \in K_v$) implies that $z^* \in \Lambda_v = \{z^* \in \Lambda : \langle z^*, Ah + v\rangle = q_v\}$ and, consequently, that

$$\begin{aligned} a^+ &\leq \inf_z \max_{z^* \in \Lambda_v} \langle z^*, Az + y\rangle = \max_{z^* \in \Lambda_v} \inf_z \langle z^*, Az + y\rangle \\ &= \begin{cases} \max_{z^* \in \Lambda_v} \langle z^*, y\rangle, & \text{if } A^* z^* = 0, \\ -\infty, & \text{otherwise} \end{cases} \\ &= \max_{z^* \in \Omega_v} \langle z^*, y\rangle. \qquad \square \end{aligned}$$

First-order estimates. We set

$$a(x, u) = \inf_{h \in K(x)} \max_{z^* \in \Omega(x)} \left[\frac{1}{2}\mathcal{L}_{xx}(z^*, x, 0)(h, h) + \mathcal{L}_u(z^*, x, 0)u\right],$$

$$a(u) = \inf_{x \in M} a(x, u),$$

$A(x, u)$ = the set of h at which infimum in the definition of $a(x, u)$ is attained.

THEOREM 3.2. *We always have*

$$p'_+(0; u) \leq a(u) \leq q(u),$$

and for any $x \in M$ and any u,

$$a(x,u) = \inf_{h\in K(x)} \max_{z^*\in\Omega(x,h)} \mathcal{L}_u(z^*,x,0)u, \tag{3.3}$$

and either $D_+^{1/2}M(x,u) = \emptyset$ or $D_+^{1/2}M(x,u) = A(x,u)$. Moreover, if $D_+^{1/2}M(x,u) \neq \emptyset$, then for any $h \in A(x,u)$,

$$p'_+(0;u) = \max_{z^*\in\Omega(x)} \left[\frac{1}{2}\mathcal{L}_{xx}(z^*,x,0)(h,h) + \mathcal{L}_u(z^*,x,0)u\right]$$

and there is a curve $h(t) \to h$ such that $t^{-1}(f(x+\sqrt{t}h(t),tu) - \vartheta) = a(x,u)$.

(Recall that according to the definition,

$$D_+^{1/2}M(x,u) = \bigcap_{\xi>0} \limsup_{t\searrow 0} t^{-1/2}(M_{t\xi}(tu) - x).)$$

Proof. For any $x \in M$, any h, and any $t > 0$, we have

$$p(tu) \le f(x+\sqrt{t}h, tu) = g(\Phi(x+\sqrt{t}h, tu)).$$

Therefore,

$$p'_+(0;u) \le \lim_{\varepsilon\to 0} \lim_{t\searrow 0} \inf_{\|h-h'\|<\varepsilon} t^{-1}(f(x+\sqrt{t}h', tu) - f(x)).$$

We observe that

$$\Phi(x+\sqrt{t}h', tu) = \Phi(x,0) + \sqrt{t}\Phi_x(x,0)h' + t(\tfrac{1}{2}\Phi_{xx}(x,0)(h,h) + \Phi_u(x,0)u) + o(t).$$

So, replacing t by t^2 and applying Lemma 3.1 to the inequality above with $v = 0, y = \frac{1}{2}\Phi_{xx}(x,0)(h,h) + \Phi_u(x,0)u$, $A = \Phi_x(x,0)$, and $\Lambda = \partial g(\Phi(x))$ (which is possible as $g(\cdot)$ is pseudopolyhedral at $\Phi(x)$) by the assumption), we conclude that for any $x \in M$ and any $h \in K(x)$

$$p'_+(0;u) \le \max_{z^*\in\Omega(x)} \left[\frac{1}{2}\mathcal{L}_{xx}(z^*,x,0)(h,h) + \mathcal{L}_u(z^*,x,0)u\right],$$

which implies that $p'_+(0;u) \le a(u)$. The equality (3.3) follows from Lemma 3.3, proved below, and the inequality $a(u) \le q(u)$ is obvious.

To prove the second part of the theorem, we assume that there is an $h \in D_+^{1/2}M(x,u)$. Then there are sequences of $t_s \to 0$ and $h_s \to h$ such that

$$f(x+\sqrt{t_s}h_s, t_s u) \le p(t_s u) + o(t_s).$$

It follows that for any $z^* \in \partial g(\Phi(x))$

$$\sqrt{t_s}\mathcal{L}_x(z^*,x,0)h_s + t_s\left[\frac{1}{2}\mathcal{L}_{xx}(z^*,x,0)(h_s,h_s) + \mathcal{L}_u(z^*,x,0)u\right] \le p(t_s u) - \vartheta + o(t_s).$$

Dividing by $\sqrt{t_s}$ and taking the limit as $S \to \infty$, we see that $\mathcal{L}_x(z^*,x,0)h \le 0$ for all $z^* \in \partial g(\Phi(x))$, which means that $h \in K(x)$. On the other hand, if $z^* \in \Omega(x)$, then dividing the inequality by t_s and taking the limit, we arrive at the inequality

$$a(x,u) \le \max_{z^*\in\Omega(x)} \left[\frac{1}{2}\mathcal{L}_{xx}(z^*,x,0)(h,h) + \mathcal{L}_u(z^*,x,0)u\right] \le p'_+(0,u),$$

which, by what has been already proved, may hold only as an equality.

In turn, together with the inequality above, this implies that

$$t_s^{-1}(p(t_s u) - \vartheta) \to p'_+(0; u).$$

Now, if e is another element of $A(x, u)$, then using Lemma 3.1 with the same A, Λ, v, and y as above, we may find a curve $e(t) \to e$ as t goes to zero such that

$$f(x + \sqrt{t}e(t), tu) - \vartheta = ta(x, u) + o(t) = tp'_+(0; u) + o(t),$$

which shows that $e \in D_+^{1/2}(x, u)$. □

The following lemma proves (3.3).

LEMMA 3.3. *Suppose that $K \subset X$ is a cone. Then for any $v \in Z$*

$$\inf_{h \in K} \max_{z^* \in \Omega(x)} \left(\frac{1}{2}\mathcal{L}''(z^*, x)(h, h) + \langle z^*, v \rangle \right) \geq \inf_{h \in K} \sup_{z^* \in \Omega(x,h)} \langle z^*, v \rangle,$$

with the equality valid if $\Omega(x, h)$ is nonempty for any $h \in K$.

Proof. If $\Omega(x, h)$ is nonempty, then (as K is a cone and $\Omega(x)$ is convex compact)

$$\begin{aligned}
&\inf_{h \in K} \max_{z^* \in \Omega(x)} \left(\frac{1}{2}\mathcal{L}''(z^*, x)(h, h) + \langle z^*, v \rangle \right) \\
&\quad = \inf_{h \in K} \inf_{t \geq 0} \max_{z^* \in \Omega(x)} \left(\frac{t}{2}\mathcal{L}''(z^*, x)(h, h) + \langle z^*, v \rangle \right) \\
&\quad = \inf_{h \in K} \max_{z^* \in \Omega(x)} \inf_{t \geq 0} \left(\frac{t}{2}\mathcal{L}''(z^*, x)(h, h) + \langle z^*, v \rangle \right) = \inf_{h \in K} \max_{z^* \in \Omega(x,h)} \langle z^*, v \rangle
\end{aligned}$$

because

$$\inf_{t \geq 0} t\mathcal{L}''(z^*, x)(h, h) = \begin{cases} -\infty & \text{if } z^* \notin \Omega(x, h), \\ 0 & \text{if } z^* \in \Omega(x, h). \end{cases}$$

If, on the other hand, $\Omega(x; h) = \emptyset$, then the supremum over $\Omega(x, h)$ is equal to $-\infty$ by the convention. □

Second-order estimates. Set

$$b(x; u, w) = \inf_{h \in K_u(x)} \max_{z^* \in \Omega_u(x)} [\mathcal{L}''(z^*, x, 0)((h, u), (h, u)) + 2\mathcal{L}_u(z^*, x, 0)w],$$

$B(x, u, w) =$ the set of h at which infimum in the definition of $b(x, u, w)$ is attained.

THEOREM 3.4. *For any $x \in M$, any u, and any w,*

$$p(tu + t^2 w) \leq p(0) + tq(x, u) + (t^2/2)b(x; u, w) + o(t^2)$$

and either

$$D_+M(x, u, w) = \emptyset \quad \text{or} \quad D_+M(x, u, w) = B(x, u, w).$$

Moreover, if $D_+M(x, u, w) \neq \emptyset$, then

$$x \in Q(u), \quad p'_+(0; u) = q(x, u), \quad p''_+(0, u, w) = b(x, u, w).$$

Proof. Fix an $x \in M$. Then for any h and any t,

$$\begin{aligned} p(tu+t^2w) &\le g(\Phi(x+th, tu+t^2w)) \\ &= p(0) + \max_{z^* \in \partial g(\Phi(x))} \langle z^*, t\Phi'(x,0)(h,u) \\ &\qquad + (t^2/2)(\Phi''(x,0)((h,u),(h,u)) + 2\Phi_u(x,u)w)\rangle \\ &\quad + o(t^2). \end{aligned}$$

Applying Lemma 3.1 with A and Λ the same as in Theorem 3.2, $v = \Phi_u(x+th,0)u$ and $y = [\mathcal{L}''(z^*,x,0)((h,u),(h,u)) + 2\mathcal{L}_u(z^*,x,0)w]$, we obtain the desired upper estimate for the value function.

The proof of the second part of the theorem is also similar to that of Theorem 3.2. Assume that there is an $h \in D_+M(x,u,w)$; hence there are sequences of $t_s \to 0$ and $h_s \to h$ such that

$$f(x+t_sh_s, t_su + t_s^2w) \le p(t_su+t_s^2w) + o(t_s^2).$$

By the first part of the theorem this means that for any $z^* \in \partial g(\Phi(x))$

$$\begin{aligned} &\langle z^*, \Phi'(x,0)(h,u) + (t_s/2)(\Phi''(x,0)((h,u),(h,u)) + 2\Phi_u(x,u)w)\rangle \\ &\quad \le t_s^{-1}[p(t_su+t_s^2w) - p(0)] \le q(x,u) + (t_s/2)b(x;u;w) + \xi t_s + o(t_s). \end{aligned} \tag{3.4}$$

The inequality implies that

$$\text{(a)} \qquad \mathcal{L}'(z^*,x,0)(h,u) \le q(x,u), \quad \forall z^* \in \partial g(\Phi(x)),$$

which means that $h \in K_u(x)$;

$$\text{(b)} \qquad \mathcal{L}_u(z^*,x,0)u \le p'_+(0;u), \quad \forall z^* \in \Omega(x),$$

which (by virtue of Corollary 3.4) implies that $p'_+(0;u) = q(x,u)$ and this in turn, together with (3.4), gives

$$\text{(c)} \qquad \mathcal{L}''(z^*,x,0)(h,u),(h,u) + 2\mathcal{L}_u(z^*,x,0) \le p''_+(0,u,w), \quad \forall\, z^* \in \Omega_u(x),$$

which may only be true if $h \in B(x,u,w)$ and $p''_+(0,u,w) = b(x,u,w)$.

In particular, we derive from this and (3.4) that, along this sequence of $t_s \to 0$,

$$p(t_su+t_s^2w) - p(0) = t_sq(x,u) + (t_s^2/2)b(x;u;w) + o(t_s^2). \tag{3.5}$$

On the other hand, let e be any element of $B(x,u,w)$. As in the proof of Theorem 3.2 it follows from Lemma 3.1 (with the same A, Λ, v, and y as earlier in this proof, but with h replaced by e) that there is a trajectory $e(t) \to e$ such that

$$f(x+te(t), tu+t^2w) \le f(x,0) + tq(x,u) + (t^2/2)b(x;u;w) + o(t^2),$$

so that $x + te(t) \in M_{o(t)}(tu+t^2w)$, whence $e \in D_+M(x,u,w)$. □

Estimates under the quadratic growth condition. Lower estimates for the value function and the solution map are more complicated and require the above-mentioned growth conditions closely connected with second-order sufficient conditions. Moreover, in certain infinite-dimensional problems, e.g., in the calculus of variations and optimal control, these growth conditions can only be stated in terms of a norm that is weaker than the norm

of the space in which the problem is stated. Therefore, to develop a flexible mechanism of obtaining lower estimates, we shall consider another norm $|\cdot|$ in X, which is *equivalent to or weaker than* $\|\cdot\|$ and compatible with the problem in the following sense:

(CC) (*compatibility condition*). There exists $k > 0$ such that for all $x \in M, h \in X$, and $u \in U$,

$$\|\Phi_x(x,0)h\| \le k|h|; \quad \|\Phi_{xx}(x,0)(h,h)\| \le k|h|\,|h|; \quad \|\Phi_{xu}(x,0)(h,u)\| \le k|h|\,|u|$$

and the second-order expansion

$$\Phi(x+h,u) = \Phi(x,0) + \Phi_x(x,0)h + \Phi_u(x,0)u + \tfrac{1}{2}\Phi''(x,0)((h,u),(h,u)) + r(x,h,u)$$

is valid with

$$\sup_{x\in M}(|h|^2 + \|u\|^2)^{-1} r(x,h,u) \to 0 \quad \text{as} \quad |h| \to 0, \quad \|u\| \to 0.$$

We observe that if the norms are equivalent, the three inequalities follow automatically from the fact that Φ is twice Fréchet differentiable and the last condition means that Φ is differentiable uniformly on M.

The following *quadratic growth* (QG) condition plays a fundamental role:

(QG) There are a $\beta > 0$ and a norm $|\cdot|$ in X that is equivalent to or weaker than $\|\cdot\|$, satisfies (CC), and such that

$$f(x) \ge \vartheta + \beta \cdot |M - x|^2, \quad \forall\, x \in D.$$

A simple and important example is provided by functionals of the calculus of variations. In this case, say, $X = C^1[0,1]$ with the norm

$$\|x(\cdot)\| = \max_{0\le t\le 1} |\dot{x}(t)| + |x(0)|$$

and the $W_{2,1}$-norm

$$|x(\cdot)| = |x(0)| + (\int |x(t)|^2 dt)^{1/2}$$

satisfies the compatibility condition under the usual assumptions on the functional. For isolated minima of functionals of the calculus of variations, (QG) is satisfied with the $|\cdot|$-norm, provided the Jacobi second-order sufficient condition is satisfied, but (QG) typically is not valid for the original C^1-norm.

In what follows we often adopt the following *uniform Lipschitzness* condition:

(UL) The function $f(x,u)$ is Lipschitz continuous in u near zero uniformly with respect to $x \in D$.

Hölder stability of approximate solutions. For any $\varepsilon \ge 0$ we set

$$\sigma_\varepsilon(u) = \sigma(M, M_\varepsilon(u)),$$

where

$$\sigma(C,D) = \sup\{|C - x| : x \in D\}$$

is the *excess* of D over C with respect to the $|\cdot|$-norm in X.

THEOREM 3.5. *Assume that* (QG) *and* (UL) *are valid. Then for any* $r > 0$ *and any function* $\varepsilon(u), 0 \le \varepsilon(u) \le r\|u\|$,

$$\limsup_{u\to 0} \sigma^2_{\varepsilon(u)}(u)/\|u\| \le (2N + r)/\beta,$$

where β *is from* (QG) *and* N *is the Lipschitz constant of* $f(x, \cdot)$.

Proof. It follows from (UL) that $p(\cdot)$ is Lipschitz near 0 with a certain constant N, the same as for $f(x, \cdot)$. For sufficiently small u and u' we thus have

$$p(u') = \min_x f(x, u') \le \min_x f(x, u) + N\|u - u'\| \le p(u) + N\|u - u'\|.$$

Assume now that sequences of $u_s \to 0$ and $\varepsilon_s \to 0$ are given such that $0 \le \varepsilon_s \le r\|u_s\|$. Then by (QG), for any sequence of $x_s \in M_{\varepsilon_s}(u_s)$ we have

$$\begin{aligned}\beta|M - x_s|^2 &\le f(x_s) - \vartheta \le f(x_s, u_s) + N\|u_s\| - p(0)\\ &\le p(u_s) - p(0) + N\|u_s\| + \varepsilon(u_s) \le (2N + r)\|u_s\|. \qquad \square\end{aligned}$$

We observe that under (QG), the excess of $M_{\varepsilon(u)}(u)$ over M with respect to the original $\|\cdot\|$-norm may not converge to zero as $u \to 0$.

A general lower estimate. Let us agree to write $x \to M$ if $|M - x| \to 0$ and denote

$$\begin{aligned}\mathcal{M}(y, \varepsilon, t) &= \{y \in M: |x - y| \le t/\varepsilon\};\\ \Omega_\delta(x) &= \{z^* \in \partial g(x): \|\mathcal{L}_x(z^*, x, 0)\| \le \delta\}.\end{aligned}$$

We set further

$$\begin{aligned}a_1(u, \varepsilon, \delta) = \liminf_{\substack{y\to M\\ t\searrow 0}} t^{-2}\Big\{&\sup_{x\in\mathcal{M}(y,\varepsilon,t)} \max_{z^*\in\Omega_\delta(x)} [\mathcal{L}_x(z^*, x, 0)(y - x)\\ &+ (1/2)\mathcal{L}_{xx}(z^*, x, 0)(y - x, y - x) + t^2\mathcal{L}_u(z^*, x, 0)u]\Big\},\\ a_2(u, \varepsilon) = \liminf_{\substack{y\to M\\ t\searrow 0}} t^{-2}\Big\{&\sup_{x\in\mathcal{M}(y,\varepsilon,t)} \max_{z^*\in\Omega(x)} [(1/2)\mathcal{L}_{xx}(z^*, x, 0)(y - x, y - x)\\ &+ t^2\mathcal{L}_u(z^*, x, 0)u]\Big\};\\ a_3(u, \varepsilon, R) = \inf_{x\in M} \inf_{\substack{h\in K_\varepsilon(x)\\ |h|\le R}} &\max_{z^*\in\Omega(x)} [(1/2)\mathcal{L}_{xx}(z^*, x, 0)(h, h) + \mathcal{L}_u(z^*, x, 0)u];\\ a_4(u, \varepsilon) = \inf_{x\in M} \inf_{h\in K_\varepsilon(x)} &\max_{z^*\in\Omega(x;h)} \mathcal{L}_u(z^*, x, 0)u.\end{aligned}$$

THEOREM 3.6. *We posit the assumptions of Theorem* 3.5. *Then for any* $\varepsilon > 0, \delta \ge 0$

$$p'_-(0; u) \ge a_1(u, \varepsilon, \delta).$$

Proof. Take a $y(t) \in D$ such that

$$f(y(t), tu) \le p(tu) + o(t).$$

By Theorem 3.5, $y(t) \to M$, so that $\mathcal{M}(y(t), \sqrt{\varepsilon}, \sqrt{t}) \ne \emptyset$ if ε is sufficiently small. For any $x \in \mathcal{M}(y(t), \sqrt{\varepsilon}, \sqrt{t})$ we have, owing to (CC):

$$\Phi(y(t), tu) = \Phi(x, 0) + \Phi_x(x, 0)h + (1/2)\Phi_{xx}(x, 0)(h, h) + t\Phi_u(x, 0)u + o(t),$$

where $h = y(t) - x$; hence, for any $z^* \in \partial g(\Phi(x,0))$

$$p(tu) - p(0) \geq \langle z^*, \Phi_x(x,0)(y(t)-x) + (1/2)\Phi_{xx}(x,0)(y(t)-x, y(t)-x) + t\Phi_u(x,0)u\rangle + o(t), \tag{3.6}$$

which implies that

$$p'_-(0;u) \geq \lim_{t\searrow 0} t^{-1}\left\{ \sup_{x\in\mathcal{M}(y(t),\sqrt{\varepsilon},\sqrt{t})} \max_{z^*\in\Omega_\delta(x)} [\mathcal{L}_x(z^*,x,0)(y(t)-x) + (1/2)\mathcal{L}_{xx}(z^*,x,0)(y(t)-x, y(t)-x) + t\mathcal{L}_u(z^*,x,0)u]\right\}.$$

Replacing t by t^2 and ε by ε^2, we obtain the desired inequality.

Simplified estimates. Theorem 3.6 is not convenient for practical calculations. We give some corollaries that offer simpler (though perhaps less precise) formulae.

THEOREM 3.7. *Under the assumptions of Theorem* 3.6, *for any sufficiently small* $\varepsilon > 0$

$$p'_-(0;u) \geq a_2(u,\varepsilon).$$

This is just a particular case of the theorem for $\delta = 0$.

THEOREM 3.8. *Under the assumptions of Theorem* 3.6 *there is a sufficiently large* $R > 0$ *such that for any* $\varepsilon > 0$

$$p'_-(0;u) \geq a_3(u,\varepsilon,R).$$

Proof. There is nothing to prove if $p'_-(0;u) = \infty$, so we assume that $p'_-(0;u) < \infty$. Let $y(t)$ be as in the proof of Theorem 3.6. Then by Theorem 3.5 there are an $R > 0$ and $x(t) \in M$ such that $y(t) = x(t) + \sqrt{t}h(t)$ with $|h(t)| \leq R\|u\|$. Therefore, we can rewrite (3.6) as follows

$$p(tu) - p(0) \geq \langle z^*, \sqrt{t}\Phi_x(x(t),0)h(t) + (t/2)\Phi_{xx}(x(t),0)(h(t),h(t)) + t\Phi_u(x(t),0)u\rangle + O(t), \tag{3.7}$$

so that for $z^* \in \Omega(x(t))$ we have

$$p(tu) - p(0) \geq t\left[\frac{1}{2}\mathcal{L}_{xx}(z^*,x(t),0)(h(t),h(t)) + \mathcal{L}_u(z^*,x(t),0)u\right] + O(t). \tag{3.8}$$

As (3.7) is valid for all $z^* \in \partial g(\Phi(x(t)))$ and $h(t)$ is bounded in the $|\cdot|$-norm, we conclude, using (CC), that

$$\limsup_{t\to 0} \max_{z^*\in\partial g(y(t))} \langle z^*, \Phi_x(x(t),0)h(t)\rangle \leq 0$$

and consequently,

$$\limsup_{t\to 0} g'(y(t);\Phi_x(x(t),0),h(t)) \leq 0. \tag{3.9}$$

If $|h(t)| \to 0$ along a sequence of t, then (3.8) remains valid with zero instead of $h(t)$ (by (CC)). Otherwise, by (3.9), $h(t) \in K_\varepsilon(x(t))$ for sufficiently small t. Along with the fact that (3.8) is valid for all $z^* \in \Omega(x(t))$, this implies the desired estimate.

Applying to this Lemma 3.3 we also get the following corollary.

COROLLARY 3.9. *Under the same assumptions,*

$$p'_-(0;u) \geq a_4(u,\varepsilon).$$

A sufficient condition for directional differentiability of the value function. A comparison of Theorem 3.2, on one hand, and Theorem 3.8 and Corollary 3.9, on the other hand, shows that either of the limit relations

$$\lim_{\varepsilon\to 0} a_3(u,\varepsilon,\infty) = a(u), \qquad \lim_{\varepsilon\to 0} a_4(u,\varepsilon) = a(u)$$

ensures the existence of the directional derivative of $p(\cdot)$ along u. The first of them will obviously be valid if we find a function ϕ from R_+ into R_+ such that $\phi(t) \to 0$ as $t \to 0$ and

$$\|K(x) - h\| \le \phi[g'(\Phi(x); \Phi_x(x,0)h)], \quad \forall\, x \in M, \quad \forall h \in X.$$

THEOREM 3.10. *We posit the assumptions of Theorem* 3.6 *and assume in addition that there are constants* $L > c > 0$ *such that*

(i) $\|\Phi_x^*(x,0)z^*\| \le L, \quad \forall\, x \in M, \quad \forall\, z^* \in \partial g(\Phi(x)),$

(ii) $\|\Phi_x^*(x,0)z^*\| \ge c, \quad \forall x \in M, \quad \forall\, z^* \in \partial g(\Phi(x))$
such that $g'(\Phi(x); \Phi_x(x,0)h) = \langle z^*, \Phi_x(x,0)h)\rangle$ *for some* $h \notin K(x)$.

Then $p(\cdot)$ *is differentiable at* 0 *along every direction and*

$$p'(0;u) = a(u).$$

Proof. Fix an $x \in M$ and consider the function $s(h) = g'(\Phi(x);\ \Phi_x(x,0)h)$. We have $s(h) \ge 0$ (as $\Omega(x) \ne \emptyset$), $K(x) = \{h\colon s(h) = 0\}$, and $s(h) \le L\|h\|$. Furthermore, for any h and any $x^* \in \partial s(h)$ there is a $z^* \in \partial g(\Phi(x))$ such that $x^* = \Phi_x^*(x,0)z^*$. Therefore, $\|x^*\| \ge c > 0$ whenever $x^* \in \partial s(h), h \ne K(x)$. It follows from [34] that

$$\|K(x) - h\| \le (L/c)s(h)$$

for all h of a neighborhood of zero and hence for all h due to the homogeneity of functions in both sides of the inequality. □

The first condition (i) in the theorem is a simple "uniform boundedness" condition both for the Lipschitz constant of $g(\cdot)$ and for the norm of $\Phi_x(x,0)$. The second is a regularity condition depending on mutual geometry of $\partial g(\Phi(x))$ and the image of $\Phi_x(x,0)$. It needs a transversality or a polyhedrality condition to be valid, or a combination of both. We shall see that it is typically valid in situations arising from constrained minimization problems like (**P**).

Estimates under the strong quadratic growth condition. We shall say that the *strong quadratic growth condition* is satisfied along u if

(SQG) There are a $\beta > 0$ and a norm $|\cdot|$ in X that are equivalent or weaker than $\|\cdot\|$, satisfy (CC), and such that

$$f(x, tu) - \vartheta \ge tq(u) + \beta|M - x|^2 + O(t^2), \quad \forall\, x \in D.$$

We shall consider this condition along with another one. Namely, we shall say that the perturbation u is *tame* if

(T) $\lim_{\varepsilon\to 0} \inf_{x\in M} \inf\{|h|\colon h \in K_{\varepsilon u}(x)\} < \infty$.

Observe that this condition is satisfied if there is an $x \in Q(u)$ such that $K_u(x) \ne \emptyset$, in particular when Z is finite dimensional, M is a finite set, and g is polyhedral.

Lipschitz stability of approximate solutions. Under (SQG) a stronger convergence property is valid for solutions.

THEOREM 3.11. *We assume* (SQG), (T), *and* (UL). *Then for any* $w \in U$ *and any nonnegative* $\xi(t) = 0(t^2)$,

$$\limsup_{t \searrow 0} t^{-1} \sigma_{\xi(t)}(tu + t^2 w) < \infty.$$

Proof. Suppose there is a $y(t) \in M_{\xi(t)}(tu + tw^2)$ such that $\tau(t) = |M - y(t)|$ goes to zero more slowly than t:

$$\limsup \tau(t)/t = \infty. \tag{3.10}$$

Taking an $r > t^{-2}\xi(t) + L\|w\|$, where L is the Lipschitz constant of $p(\cdot)$, we obtain the following estimate using (SQG):

$$\begin{aligned} p(tu + t^2 w) - p(0) - tq(u) &\geq p(tu) - p(0) - tq(u) - t^2 L\|w\| \\ &\geq f(y(t), tu) - p(0) - tq(u) - \xi(t) - t^2 L\|w\| \\ &\geq f(y(t), tu) - p(0) - tq(u) - rt^2 \geq \beta\tau^2(t) + 0(t^2) = 0(\tau^2(t)). \end{aligned}$$

On the other hand, fix an $x(t) \in M$ with $q(x(t), u) \leq q(u) + t^2$ such that (T) holds. Then there are a constant N and $h(t)$ such that $|h(t)| \leq N$ for all t and $g'(\Phi(x(t)); \Phi'(x(t), 0)(h(t), u)) \leq q(x(t), u) + t^2$, and we have, owing to (PP),

$$\begin{aligned} p(tu + t^2 w) - p(0) - tq(u) &\leq L\|w\|t^2 + p(tu) - p(0) - tq(u) \\ &\leq g(\Phi(x(t) + th(t), tu)) - g(\Phi(x(t), 0)) - tq(u) + 0(t^2) \\ &= t[g'(\Phi(x(t)); \Phi'(x, 0)(h(t), u)) - q(u)] + 0(t^2) = 0(t^2). \end{aligned}$$

This contradicts the previous inequality in view of (3.10). □

First-order directional derivative. As an immediate corollary we obtain the existence of the local directional derivative of the value function.

THEOREM 3.12. *Under the assumptions of Theorem* 3.11 *the first-order directional derivatives of* $p(\cdot)$ *along* u *exists and is equal to*

$$p'(0; u) = q(u) = \inf_{x \in M} \max_{z^* \in \Omega(x)} \mathcal{L}_u(z^*, x, 0)u.$$

Proof. We observe that $\Omega(x, 0) = \Omega(x)$, so the estimate

$$p'_+(0; u) \leq \max_{z^* \in \Omega(x)} \mathcal{L}_u(z^*, x, 0)u, \quad \forall x \in S$$

follows from Theorem 3.2.

On the other hand, if $y(t) \in M_{\xi(t)}(tu)$, where $\xi(t) = 0(t^2)$, then $y(t) = x(t) + te(t)$, where $x(t)$ belongs to M and $\|e(t)\| \leq r < \infty$. So setting $h(t) = \sqrt{t}e(t)$, we deduce from (3.8) that for any $z^* \in \Omega(x(t))$

$$p(tu) - p(0) \geq t\mathcal{L}_u(z^*, x(t), 0)u + o(t),$$

and consequently, that

$$p'_-(0; u) \geq \inf_{x \in S} \max_{z^* \in \Omega(x)} \mathcal{L}_u(\lambda, x, 0)u. \qquad \square$$

Second-order differentiation of the value function. In principle, it is possible to write a general second-order lower estimate similar in spirit to that of Theorem 3.6. But it will look rather awkward and we shall give only second-order counterparts of Theorems 3.7–3.9.

We set

$$b_1(u,w,\varepsilon)=\liminf_{\substack{y\to M\\ t\searrow 0}}\sup_{x\in\mathcal{M}(y,\varepsilon,t)}\max_{z^*\in\Omega_u(x)} t^{-2}[\mathcal{L}''(z^*,x,0)((y-x,tu),(y-x,tu))$$
$$+2t^2\mathcal{L}_u(z^*,x,0)u];$$
$$b_2(u,w,\varepsilon,r)=\inf_{x\in Q_\varepsilon(u)}\inf_{\substack{|h|\le r\\ h\in K_{\varepsilon u}(x)}}\max_{z^*\in\Omega_u(x)}[\mathcal{L}''(z^*,x,0)((h,u),(h,u))$$
$$+2\mathcal{L}_u(z^*,x,0)w].$$

THEOREM 3.13. *Under the assumptions of Theorem* 3.11, *for any* $\varepsilon>0$

$$p''_-(0;u;w)\ge b_1(u,w,\varepsilon).$$

Proof. If $y(t)\in M_{\xi(t)}(tu+t^2w)$, where $\xi(t)=0(t^2)$, then by Theorem 3.11 there is an $r>0$ such that $|M-y(t)|\le rt$. So $\mathcal{M}(y(t),\varepsilon,t)\ne\emptyset$ if ε is sufficiently small, and for any $x\in\mathcal{M}(y(t),\varepsilon,t)$ we have

$$\Phi(y(t),tu+t^2w)=\Phi(x,0)+\Phi'(x,0)(y(t)-x,tu)$$
$$+\tfrac12\Phi''(x,0)((y(t)-x,tu),(y(t)-x,tu))+t^2\Phi_u(x,0)w+o(t^2),$$

so for $z^*\in\Omega_u(x)$ we have

$$p(tu+t^2w)-p(0)\ge tq(u)+\tfrac12\mathcal{L}''(z^*,x,0)((y(t)-x,tu),(y(t)-x,tu))$$
$$+t^2\mathcal{L}_u(z^*,x,0)w+o(t^2).\qquad\square$$

THEOREM 3.14. *We posit the assumptions of Theorem* 3.11. *Then there is an* $r>0$ *such that for any* $\varepsilon>0$

$$p''_-(0;u;w)\ge b_2(u,w,\varepsilon,r).$$

Proof. Let $y(t)$ and $r>0$ be as in the proof of Theorem 3.13. Take an $x(t)\in M$ with $|y(t)-x(t)|\le rt$ and define $h(t)$ by $y(t)=x(t)+th(t)$. Since $g(\cdot)$ is uniformly pseudopolyhedral,

$$\begin{aligned}(p(tu+t^2w)-p(0)&\ge g'(\Phi(x(t);t\Phi'(x(t),0)(h(t),u))\\ &+(t^2/2)\Phi''(x(t);u)(h(t),u)(h(t),u))+t^2\Phi_u(x(t),0)w)+o(t^2)\\ &=tg'(\Phi(x(t));\Phi'(x(t),0)(h(t),u))+0(t^2).\end{aligned}\tag{3.11}$$

If $p''_-(0;u;w)<\infty$, this implies that

$$\liminf_{t\searrow 0}t^{-1}[g'(\Phi(x(t));\Phi_x(x(t),0)h(t)+\Phi_u(x(t),0)u)-q(u)]\le 0.\tag{3.12}$$

If $u=0$ and $|h(t)|\to 0$, nothing will change if we replace $h(t)$ by zero in (3.11). Otherwise (3.12) implies that $h(t)\in K_{\varepsilon u}(x(t))$ along a sequence of t's for any $\varepsilon>0$. In either case we obtain from (3.11)

$$p(tu+tw^2)-p(0)-tq(u)\ge\frac{t^2}{2}\inf_{\substack{|h|\le r\\ h\in K_{\varepsilon u}(x(t))}}\max_{z^*\in\Omega_u(x(t))}[\mathcal{L}''(z^*,x,0)((h(t),u),(h(t),u))$$
$$+2\mathcal{L}_u(z^*,x,0)w]+o(t^2).$$

It remains to observe that (3.12) also implies that

$$q(x(t),u) = \max_{z^* \in \Omega(x(t))} \mathcal{L}_u(z^*, x(t), 0)u$$
$$= \max_{z^* \in \Omega(x(t))} [\mathcal{L}_x(z^*, x(t), 0)h(t) + \mathcal{L}_u(z^*, x(t), 0)u] \le q(u) + o(1),$$

which shows that $x(t) \in Q_\varepsilon(u)$ if t is sufficiently small. □

The following theorem, similar in spirit to Theorem 3.10, gives a sufficient condition for the existence of the second-order directional derivative of the value function along the given direction u.

THEOREM 3.15. *Assume that*

(i) M *is a finite set,*

(ii) (SQG) *holds,*

(iii) $K_u(x) \not\equiv \emptyset$ *on* $Q(u)$,

(iv) *there is a* $c > 0$ *such that* $\|\Phi_x^*(x,0)z^*\| \ge c$ *whenever* $x \in Q(u)$, $z^* \in \partial g(\Phi(x))$, *and* $\langle z^*, \Phi'(x,0)(h,tu)\rangle = g'(\Phi(x); \Phi'(x,0)(h,tu))$ *for some* $h \notin K_{tu}(x)$, $t \ge 0$.

Then $p(\cdot)$ *is twice differentiable along* u *and for any* w

$$p''(0;u;w) = \inf_{x \in Q(u)} b(x,u,w).$$

Proof. First we observe that, owing to (i), $Q(u) \ne \emptyset$, $Q(u) = Q_\varepsilon(u)$ for sufficiently small ε, and (UL) holds; also (i) and (iii) together imply (T). Thus, all of the conditions of Theorem 3.11 are satisfied. Finally, consider the function $s(h,t) = g'(\Phi(x);\ \Phi'(x,0)(h,tu)) - tq(x,u)$. It follows from (iv), as in the proof of Theorem 3.10, that for any $(x^*, \alpha) \in \partial s(h,t)$ we have $\|(x^*,\alpha)\| \ge c$, provided $h \notin K_{tu}(x)$, and we get the desired result in a similar way, taking into account that $s(h,t) \ge 0$ for all (h,t) and $s(h,t) = 0$ if and only if $h \in K_{tu}(x)$.

Estimates under the linear growth condition. We shall say that a *linear growth condition* is satisfied if

(LG) There is a $\beta > 0$ such that

$$f(x) \ge \vartheta + \beta|M - x|, \quad \forall x \in D,$$

where $|\cdot|$ is a norm with the same properties as in (QG) and (SQG).

Remark. It is of course possible to consider growth conditions with arbitrary positive exponent α. It seems, however, that only the cases $\alpha = 2$ and $\alpha = 1$ are really important.

The same arguments as in the proof of Theorem 3.5 prove the following.

THEOREM 3.16. *We assume* (LG) *and* (UL). *Then for any* $r > 0$ *and any* $\varepsilon(u), 0 \le \varepsilon(u) \le r\|u\|$

$$\limsup \sigma_{\varepsilon(u)}(u)/\|u\| \le (2N + r)/\beta,$$

where β *is from* (LG) *and* N *is the Lipschitz constant of* $f(x,\cdot)$.

As an easy consequence of the theorem we get the following result.

THEOREM 3.17. *Under the assumptions of Theorem* 3.16 $p'(0;u)$ *exists for any* u *and*

$$p'(0,u) = q(u).$$

Moreover, $t^{-1}(p(tu) - p(0))$ *converges to* $q(u)$ *uniformly on bounded subsets of* U, *provided* $\Omega(x)$ *is norm bounded on* M.

Proof. The proof is similar to that of Theorem 3.12. If $y = y(t) \in M_{\xi(t)}(tu)$ and $\xi(t) = 0(t), \|u\| \le c < \infty$, then by Theorem 3.16 $y(t) = x(t) + te(t)$ with $x(t) \in M, \|e(t)\| \le (N+t)c/\beta = N_1$. So setting $h(t) = \sqrt{t}e(t)$ and applying (3.8) and (CC), we get

$$t^{-1}(p(tu) - p(0)) \ge q(u) - tN_1c\gamma,$$

where γ is a bound for the norms of $z^* \in \Omega(x), x \in M$. □

We leave to the reader the extension of Theorems 3.13–3.15 to the case when (SQG) is replaced by (LG). This extension is actually straightforward, for all that is needed is the replacement of the reference to Theorem 3.11 by the reference to Theorem 3.16.

4. Optimization under constraints. We return to the study of the constrained problem $(\mathbf{P}_\varepsilon(u))$. The Lagrangian of the problem is

$$\mathcal{L}(z^*, x, u) = \sum_{i=0}^{m} \lambda_i f_i(x, u) + (y^* \circ F)(x, u).$$

where we set $z^* = (\lambda_o, \dots, \lambda_m, y^*)$, and the Lagrangian of the associated composite optimization problem $(\mathbf{C}(\mu, u))$ is

$$\mathcal{L}(z^*, x, u) - \lambda_o\mu. \tag{4.1}$$

For the set $\Omega(x)$ of Lagrange multipliers of the unperturbed problem (**C**) in terms of the data of (**P**), we have the expression

$$\Omega(x) = \Big\{z^*: \lambda_i \ge 0, i = 0, \dots, m; \lambda_i = 0, i \notin I(x); \sum \lambda_i + \|y^*\| \le 1; \mathcal{L}_x(z^*, x, 0) = 0\Big\},$$

where

$$I(x) = \{0\} \cup \{i: f_i(x, 0) = 0\}$$

is the set of *active indices*. We denote by $\Lambda(x)$ the standard set of Lagrange multipliers for (**P**):

$$\Lambda(x) = \{z^*: \lambda_i \ge 0, i = 0, \dots, m; \lambda_i = 0, i \notin I(x); \mathcal{L}_x(z^*, x, 0) = 0\},$$

and by

$$\Lambda_1(x) = \{z^* \in \Lambda(x): \lambda_o = 1\}, \qquad \Lambda_o(x) = \{z^* \in \Lambda(x): \lambda_o = 0\}$$

the sets of *normal* and *abnormal* multipliers. Observe that $\Lambda_o(x)$ is the recession cone of $\Lambda_1(x)$, on one hand, and the cone generated by

$$\Omega_o(x) = \Omega(x) \cap \Lambda_o(x) = \{z^* \in \Omega(x): \lambda_o = 0\},$$

on the other. The expression for the critical cone is specified as follows:

$$K(x) = \{h \in X: f_{ix}(x, 0)h \le 0, i \in I(x); F_x(x, 0)h = 0\}$$

and

$$K_\varepsilon(x) = \{h \in X: f_{ix}(x, 0)h \le \varepsilon\|h\|, i \in I(x); \|F_x(x, 0)h\| \le \varepsilon\|h\|\}.$$

Reduction technique for estimates. The reduction theorems of §2 give formulae connecting directional derivatives and solution sets of the associated constrained and composite optimization problems. We begin here by modifying the formulae to obtain the corresponding relations for the estimates.

It can be observed that the first-order estimates found in the preceding section have a common structure that can be described in terms of quadruples $\{\mathfrak{X}, \Psi, \Gamma, \mathfrak{F}\}$, where $\mathfrak{X}$ is a set, $\Psi(z^*, \mathfrak{x}, u)$ a function on $Z^* \times \mathfrak{X}$, $\Gamma(\mathfrak{x})$ a set-valued map from $\mathfrak{X}$ into Z^* with convex and weak* compact values, and $\mathfrak{F}$ a nondecreasing functional defined on extended-real-valued functions on $\mathfrak{X}$.

The estimates themselves have the form

$$e(u) = \mathfrak{F}[\max_{z^* \in \Gamma(\mathfrak{x})} \Psi(z^*, \mathfrak{x}, u)]. \tag{4.2}$$

More specifically:

$\mathfrak{X}$ is a subset of the product of M and a certain set of other variables (e.g., $y, h, t, \delta, \varepsilon$);

$\Psi(z^*, \mathfrak{x}, u)$ is a combination of derivatives of the Lagrangian, in particular, it is a linear continuous function of z^*;

$\Gamma(\mathfrak{x})$ is a subset of the subdifferential of the function $g(\cdot)$ at a certain point; and

$\mathfrak{F}$ is a finite sequence of maximizations and minimizations over components of $\mathfrak{x}$.

For example, the estimate $a(u)$ of Theorem 3.2 corresponds to

$$\mathfrak{X} = \{(x, h): x \in M, h \in K(x)\},$$

$$\begin{cases} \Gamma(x, h) = \Omega(x), \\ \Psi(z^*, (x, h), u) = (1/2)\mathcal{L}_{xx}(z^*, x, 0)(h, h) + \mathcal{L}_u(z^*, x, 0)u, \end{cases} \tag{4.3}$$

$$\mathfrak{F}(\Psi(\cdot)) = \inf_{x \in M} \inf_{h \in K(x)} \Psi(x, h).$$

Applying this to the problem $(\mathbf{C}(\mu, u))$, we have to replace u by (μ, u) and $\mathcal{L}_u(z^*, x, 0)u$ by $-\lambda_o \mu + \mathcal{L}_u(z^*, x, 0)u$. Therefore, the estimate specifies for (**C**) as follows:

$$e(\mu, u) = \mathfrak{F}[\max_{z^* \in \Gamma(\mathfrak{x})} (-\lambda_o \mu + \Psi(z^*, \mathfrak{x}, u))]. \tag{4.4}$$

By Theorem 2.1 the corresponding estimate for the variational derivative of the value function in (**P**) has the form

$$\mathbf{e}(u) = \inf\{\mu: e(\mu, u) \leq 0\}. \tag{4.5}$$

Furthermore, the following simple fact holds.

PROPOSITION 4.1. *Let* $\Gamma(\mathfrak{x})$ *be a map into* $R_+^{m+1} \times Y^*$. *Denote by* $\Delta(\mathfrak{x})$ *the cone generated by* $\Gamma(\mathfrak{x})$ *and set*

$$\Delta_1(\mathfrak{x}) = \{z^* = (1, \lambda_1, \ldots, \lambda_m, y^*) \in \Delta(\mathfrak{x})\},$$
$$\Gamma_o(\mathfrak{x}) = \{z^* = (0, \lambda_1, \ldots, \lambda_m, y^*) \in \Gamma(\mathfrak{x})\}.$$

Then

$$\mathbf{e}(u) = \mathfrak{F}[\sup_{z^* \in \Delta_1(\mathfrak{x})} \Psi(z^*, \mathfrak{x}, u)], \tag{4.6}$$

provided the following condition holds:

$$\mathfrak{F}[\max_{z^*\in\Gamma(\mathfrak{x})}\Psi(z^*,\mathfrak{x},0)]\le 0,\quad \text{if } \Gamma(\mathfrak{x})=\Gamma_o(\mathfrak{x}). \tag{4.7}$$

As usual we adopt the standard convention $\sup\emptyset=-\infty$.

Proof. It is clear that $\Gamma(\mathfrak{x})=\Gamma_o(\mathfrak{x})$ if and only if $\Delta_1(\mathfrak{x})=\emptyset$. Therefore, if $\Gamma(\mathfrak{x})=\Gamma_o(\mathfrak{x})$ for all $\mathfrak{x}\in\mathfrak{X}$, then both (4.5) and (4.6) give $-\infty$. On the other hand, if $\Gamma_+(\mathfrak{x})=\Gamma(\mathfrak{x})\backslash\Gamma_o(\mathfrak{x})\neq\emptyset$, then $\Gamma_+(\mathfrak{x})$ is dense in $\Gamma(\mathfrak{x})$ and the maximum over the latter in (4.4) can be replaced by the supremum over the first. As the relations

$$-\lambda_o\mu+\Psi(z^*,\mathfrak{x},u)\le 0,\qquad z^*=(\lambda_o,\lambda_1,\dots,\lambda_m,y^*)\in\Gamma(\mathfrak{x}),\lambda_o>0$$

and

$$-\mu+\Psi(z^*,\mathfrak{x},u)\le 0,\qquad z^*/\lambda_o\in\Delta_1(\mathfrak{x}),\lambda_o>0$$

are equivalent, we conclude that (4.5) can be rewritten as

$$\mathbf{e}(u)=\inf\{\mu\colon\mathfrak{F}[\max_{z^*\in\Delta_1(\mathfrak{x})}(-\mu+\Psi(z^*,\mathfrak{x},u))\le 0],$$

and this is obviously the same as (4.6). □

It must be observed that (4.7) holds if $\Delta_1(\mathfrak{x})\neq\emptyset$ for all $\mathfrak{x}\in\mathfrak{X}$ and, conversely, if $|\mathbf{e}(u)|<\infty$, then $\Delta_1(\mathfrak{x})\neq\emptyset$ for those $\mathfrak{x}$ that are essential for the calculation of $a(u)$ (otherwise the quantity in the square brackets of (4.4) does not depend on μ and $\mathbf{e}(u)$ must equal to plus or minus infinity).

Proposition 4.1 and (4.5) give together a precise algorithm of translating the estimates of §3 into corresponding results for $(\mathbf{P}_\varepsilon(u))$.

Note. In what follows we often replace italic letters by corresponding boldface letters to denote objects connected with **(P)** as opposed to the corresponding objects relating to **(C)**.

Estimates for the first-order variational derivatives. Let $e(u)$ denote any of the first-order estimates of §3: $a(x,u), a(u), a_1(u,\varepsilon,\delta), a_2(u,\varepsilon), a_3(u,\varepsilon,R), a_4(u,\varepsilon), q(x,u), q(u)$, and let $e(\mu,u)$ be defined by the same formula as $e(u)$ but with $\mathcal{L}_u(z^*,x,0)u$ replaced by $-\lambda_o\mu+\mathcal{L}_u(z^*,x,0)u$, as in (4.4). We also recall that $M=S$ by Theorem 2.2, so we can replace M by S in all of the formulae. For instance,

$$\begin{aligned}
a(x,\mu,u)&=\inf_{h\in K(x)}\max_{z^*\in\Omega(x)}\left(-\lambda_o\mu+\frac{1}{2}\mathcal{L}_{xx}(z^*,x)(h,h)+\mathcal{L}_u(z^*,x,0)u\right),\\
q(x,\mu,u)&=\max_{z^*\in\Omega(x)}(-\lambda_o\mu+\mathcal{L}_u(z^*,x,0)u),\\
q(\mu,u)&=\inf_{x\in S}q(x,\mu,u),\\
a_4(\mu,u,\varepsilon)&=\inf_{x\in M}\inf_{h\in K_\varepsilon(x)}\max_{z^*\in\Omega(x;h)}(-\lambda_o\mu+\mathcal{L}_u(z^*,x,0)u),\quad\text{etc.}
\end{aligned}$$

Finally, by the corresponding boldface letter $\mathbf{e}(u)$ we denote the quantity obtained from $e(\mu,u)$ by means of (4.5) so that, say,

$$\begin{aligned}
\mathbf{a}(u)&=\inf\{\mu\colon a(\mu,u)\le 0\},\\
\mathbf{q}(u)&=\inf\{\mu\colon q(\mu,u)\le 0\},\quad\text{etc.}
\end{aligned}$$

Then Theorem 2.1 together with Theorems 3.2, 3.5–3.8, and 3.16 gives the following.

THEOREM 4.2. (a) *We always have*

$$d_+v(0;u) \le \mathbf{a}(u) \le \mathbf{q}(u);$$

(b) *if the conditions* (QG) *and* (UL) *are valid, then for any* ε, $\delta > 0$ *and sufficiently large* R,

$$d_-v(0;u) \ge \mathbf{a}^-(u),$$

where $\mathbf{a}^-(u)$ *is any of* $\mathbf{a}_1(u,\varepsilon,\delta), \mathbf{a}_2(u,\varepsilon), \mathbf{a}_3(u,\varepsilon,R), \mathbf{a}_4(u,\varepsilon)$;

(c) *if the conditions* (LG) *and* (UL) *are valid, then* $dv(0;u)$ *exists for all* u *and*

$$dv(0;u) = \mathbf{q}(u).$$

The conditions (QG), (LG), and (UL) here and below are the same as in §3, the first two stated, of course, in terms of the function

$$f(x) = \max\{f_o(x,0) - \theta, \max_{1\le i\le m} f_i(x,0), \|F(x,0)\|\};$$

and the third applied to $\Phi(x,\mu,u) = (f_o(x,u) - \mu - \theta, \dots, f_m(x,u), F(x,u))$.

Using Proposition 4.1, we obtain explicit formulae for the boldface estimates as follows.

PROPOSITION 4.3. (a) *If for an* $x \in S$ *either*

$$\Lambda_1(x) \ne \emptyset \quad \text{or} \quad \mathcal{L}_u(z^*,x,0)u \le 0, \quad \forall\, z^* \in \Lambda_o(x), \tag{4.8}$$

then

$$\mathbf{a}(x,u) = \inf_{h\in K(x)} \sup_{z^*\in\Lambda_1(x)} \left(\frac{1}{2}\mathcal{L}_{xx}(z^*,x,0)(h,h) + \mathcal{L}_u(z^*,x,0)u\right),$$

so that

$$\mathbf{a}(u) = \inf_{x\in S} \inf_{h\in K(x)} \sup_{z^*\in\Lambda_1(x)} \left(\frac{1}{2}\mathcal{L}_{xx}(z^*,x,0)(h,h) + \mathcal{L}_u(z^*,x,0)u\right);$$

$$\mathbf{a}_2(u,\varepsilon) = \liminf_{\substack{y\to M\\ t\searrow 0}} t^{-2}\left\{\sup_{x\in\mathcal{M}(y,\varepsilon,t)} \sup_{z^*\in\Lambda_1(x)} [(1/2)\mathcal{L}_{xx}(z^*,x,0)(y-x,y-x) + t^2\mathcal{L}_u(z^*,x,0)u]\right\} \tag{4.9}$$

$$\mathbf{a}_3(u,\varepsilon,R) = \inf_{x\in S} \inf_{\substack{h\in K_\varepsilon(x)\\ |h|\le R}} \sup_{z^*\in\Lambda_1(x)} [(1/2)\mathcal{L}_{xx}(z^*,x,0)(h,h) + \mathcal{L}_u(z^*,x,0)u].$$

(b) *If for an* $x \in S$ *and any* $h \in K(x)$ (*respectively*, $h \in K_\varepsilon(x)$), *either*

$$\Lambda_1(x;h) \ne \emptyset \quad \text{or} \quad \mathcal{L}_u(z^*,x,0)u \le 0, \quad \forall\, z^* \in \Omega(x,h) \quad \text{if } \Omega(x,h) \subset \Omega_o(x), \tag{4.10}$$

then

$$\mathbf{a}(u) = \inf_{x\in S} \inf_{h\in K(x)} \max_{z^*\in\Lambda_1(x;h)} \mathcal{L}_u(z^*,x,0)u;$$

$$\mathbf{a}_4(u,\varepsilon) = \inf_{x\in S} \inf_{h\in K_\varepsilon(x)} \max_{z^*\in\Lambda_1(x;h)} \mathcal{L}_u(z^*,x,0)u.$$

In either case

$$\mathbf{q}(x,u) = \sup_{z^* \in \Lambda_1(x)} \mathcal{L}_u(z^*,x,0)u. \tag{4.11}$$

Here $\Lambda_1(x;h) = \{z^*: z^* \in \Lambda(x), \mathcal{L}_{xx}(z^*,x,0)(h,h) \geq 0\}$.

The following two propositions give conditions which guarantee that the upper and lower variational derivatives of the value function are finite.

PROPOSITION 4.4. *A sufficient condition for $d_+v(0;u) < \infty$ is that there is an $x \in S$ such that*

$$\mathcal{L}_u(z^*,x,0)u \leq 0, \quad \forall z^* \in \Lambda_o(x). \tag{4.12}$$

Proof. Set

$$L(x) = \{l = (\lambda_o,\dots,\lambda_m): \lambda_i \geq 0, i = 0,\dots,m; \lambda_i = 0, i \notin I(x); \sum \lambda_i f_{ix}(x,0) \in \operatorname{Im} F_x^*(x,0)\}.$$

$$L_\alpha(x) = \{l = (\lambda_o,\dots,\lambda_m) \in L(x): \lambda_o = \alpha\}, \qquad \alpha = 0,1.$$

It is clear that $L(x)$ is a polyhedral convex cone, generated by $L_o(x) \cup L_1(x)$, so there is a finite number of elements $l_1,\dots,l_k \in L_1(x)$ such that for any $l \in L(x)$ we have a representation

$$l = \sum \alpha_j l_j + \hat{l},$$

where $\alpha_j > 0, \hat{l} \in L_o(x)$, and $\sum \alpha_j = \lambda_o$.

If now $z^* = (l,y^*) \in \Omega(x) \subseteq \Lambda(x)$ (in which case $l \in L(x)$) and $y_1^*,\dots,y_k^*$ are such that $z_j^* = (l_j, y_j^*) \in \Lambda(x)$, then there are $\alpha_j \geq 0, \hat{z}^* = (\hat{l},\hat{y}^*) \in \Lambda_o(x)$, such that $z^* = \sum \alpha_j z_j^* + \hat{z}^*, \sum \alpha_j = \lambda_o$.

Choose a $\mu \geq \max_j \mathcal{L}_u(z_j^*,x,0)u$. Then (4.12) implies that

$$\mathcal{L}_u(z^*,x,0)u - \lambda_o\mu = \mathcal{L}_u(\hat{z}^*,x,0)u + \sum \alpha_j(\mathcal{L}_u(z_j^*,x,0)u - \mu) \leq 0.$$

This is true for any $z^* \in \Omega(x)$, whence $q(x,u) \leq \mu < \infty$. □

We shall say that (**P**) is *normal* if

$$\inf_{x \in S} \max_{\lambda \in \Omega(x)} \lambda_o > 0. \tag{N}$$

PROPOSITION 4.5. *If* (QG) *and* (UL) *hold, then* (N) *is sufficient for $-\infty < d_-v(0;u)$ for every u; it is also necessary if S is a compact set and* (4.12) *is satisfied for all $x \in S$.*

Proof. If (N) holds, then there is an $N > 0$ such that

$$\max\{\lambda_o: z^* \in \Omega(x)\} \geq N^{-1}, \quad \forall x \in S.$$

By Theorem 4.2 there is an $R > 0$ such that $d_-v(0;u) \geq \mathbf{a}_3(u,\varepsilon,R)$ for any $\varepsilon > 0$. As follows from (4.5), for any $\mu > \mathbf{a}_3(u,\varepsilon,R)$ we have

$$0 \geq \inf_{x \in S} \inf_{\substack{h \in K_\varepsilon(x) \\ |h| \leq R}} \max_{z^* \in \Omega(x)} [-\lambda_o\mu + (1/2)\mathcal{L}_{xx}(z^*,x,0)(h,h) + \mathcal{L}_u(z^*,x,0)u].$$

$$\geq \inf_{x \in S} \max_{z^* \in \Omega(x)} (-\lambda_o\mu) - \frac{1}{2}kR^2 - r \geq -\mu/N - \frac{1}{2}kr^2 - r,$$

where k comes from the compatibility condition (CC) of §3 along with the facts that $|h| \le R$ and

$$r = \inf_{x \in S} \max_{z^* \in \Omega(x)} |\mathcal{L}_u(z^*, x, 0)u| < \infty.$$

It follows that any $\mu \ge \mathbf{a}_3(u, \varepsilon, R)$ also satisfies $\mu \ge -N(k + r)$; hence $\mathbf{a}_3(u, \varepsilon, R) \ge -N(k + r)$.

Assume now that S is a compact set, (4.12) is satisfied for all $x \in S$, but the normality condition (N) fails to be valid. Then there is a sequence $\{x_n\} \subset S$ such that

$$\lim_{n \to \infty} \max\{\lambda_o \colon z^* \in \Omega(x_n)\} = 0.$$

We may assume that $x_n \to x \in S$. Then (since $\Omega(\cdot)$ is norm-to-weak* upper semicontinuous set-valued mapping) lim sup $\Omega(x_n) \subset \Omega_o(x)$ and therefore

$$\limsup_{n \to \infty} \max_{\lambda \in \Omega(x_n)} (-\lambda_o \mu + \mathcal{L}_u(z^*, x, 0)u) \le 0, \quad \forall \mu.$$

Therefore, $a(\mu, u) \le 0$ for all μ and $d_- v(0; u) = -\infty$. □

Strong normality and strong quadratic growth conditions. Unlike (QG) the strong quadratic growth condition of §3 depends on perturbation and therefore needs a modification to apply to (**P**). So we shall say that (**P**) satisfies the *strong quadratic growth condition* along u if

(SQG) $|\mathbf{q}(u)| < \infty$ and (SQG) holds for (**C**) along $(\mathbf{q}(u), u)$.

The latter means that there is a norm $|\cdot|$ on X weaker or equivalent to $\|\cdot\|$ and satisfying (CC) (with Φ being replaced by $(f_o, \dots, f_m, F)$) and $\beta > 0$ such that

$$f(x, t\mathbf{q}(u), tu)) \ge \beta |S - x|^2 + O(t^2), \quad \forall x \in D. \tag{4.13}$$

We shall use (**SQG**) together with the following strengthened version of the normality condition.

We shall say that (**P**) is *strongly normal along* u if

(SN) $|\mathbf{q}(u)| < \infty$ and $\inf_{x \in S} \max\{\lambda_o \colon z^* \in \Omega_{(\mathbf{q}(u), u)}(x)\} > 0$,

where

$$\Omega_{(\mu, u)}(x) = \arg \min_{z^* \in \Omega(x)} (-\lambda_o \mu + \mathcal{L}_u(z^*, x, 0)u).$$

PROPOSITION 4.6. *If* (**SN**) *holds, then*

$$\mathbf{q}(u) = \lim_{\varepsilon \to 0} \inf_{x \in S} \mathbf{q}_\varepsilon(x, u),$$

where

$$\mathbf{q}_\varepsilon(x, u) = \inf\{\mu \colon q(x, \mu, u) \le \varepsilon\}. \tag{4.14}$$

Proof. By definition,

$$\mathbf{q}(u) = \inf\{\mu \colon q(\mu, u) \le 0\} = \inf\{\mu \colon \inf_{x \in S} q(x, \mu, u) \le 0\}.$$

Therefore, for any $\mu > \mathbf{q}(u)$ there is an $x \in S$ with $q(x, \mu, u) < \varepsilon$ so that $\mu \geq \mathbf{q}_\varepsilon(x, u)$ and, consequently,

$$\mathbf{q}(u) \geq \lim_{\varepsilon \to 0} \inf_{x \in S} \mathbf{q}_\varepsilon(x, u). \tag{4.15}$$

This inequality is valid unconditionally.

Conversely, assume (**SN**) and take a $\mu \geq \lim_{\varepsilon \to 0} \inf_{x \in S} \mathbf{q}_\varepsilon(x, u)$. Then for any $\varepsilon > 0$ there is an $x \in S$ such that

(4.16)

$$\begin{aligned} \varepsilon \geq q(x, \mu, u) &= \max_{z^* \in \Omega(x)} (-\lambda_o \mu + \mathcal{L}_u(z^*, x, 0)u) \\ &\geq -\gamma(\mu - \mathbf{q}(u)) + \max\{-\lambda_o \mathbf{q}(u) + \mathcal{L}_u(z^*, x, 0)u \colon z^* \in \Omega_{(\mathbf{q}(u), u)}(x)\} \\ &\geq -\gamma(\mu - \mathbf{q}(u)) + q(\mathbf{q}(u), u) \geq -\gamma(\mu - \mathbf{q}(u)) \end{aligned}$$

(for $q(x, \mu, u)$ is always nonnegative), where

$$\gamma = \inf_{x \in S} \max\{\lambda_o \colon z^* \in \Omega_{(\mathbf{q}(u), u)}(x)\} > 0.$$

As this is true for any $\varepsilon > 0$, we conclude that $\mu \geq \mathbf{q}(u)$. □

THEOREM 4.7. *We assume that* (**SQG**), (UL), *and* (**SN**) *hold. Then the variational derivative* $dv(0; u)$ *exists and*

$$dv(0; u) = \mathbf{q}(u).$$

Proof. In view of Theorem 4.2 we have only to prove that $d_- v(0; u)$ is not smaller than $\mathbf{q}(u)$. To this end we have to verify, according to Theorem 2.1, that $p'_-(0; (\mu, u)) > 0$ if $\mu < \mathbf{q}(u)$. Assume the contrary: $p'_-(0; (\mu, u)) = 0$ for some $\mu < \mathbf{q}(u)$. Then there are sequences of $t_n \to 0, r_n \geq 0, r_n = o(t_n)$, and $y_n \in X$ such that

$$p(t_n \mu, t_n u) \leq r_n \quad \text{and} \quad f(y_n, t_n \mu, t_n u) \leq p(t_n \mu, t_n u) + r_n.$$

Take an $x_n \in S$ such that $|x_n - y_n| \leq 2|S - y_n|$ and set $h_n = x_n - y_n$. Then by (4.13)

$$\begin{aligned} 2r_n &\geq f(x_n + h_n, t_n \mu, t_n u) \geq f(x_n + h_n, t_n \mathbf{q}(u), t_n u) \\ &\geq \beta |S - y_n|^2 + O(t_n^2) \geq (1/2)|h_n|^2 + O(t_n^2). \end{aligned}$$

It follows that $|h_n| = o(t_n^{1/2})$.

On the other hand, for any $x \in S$, any t, and any $h = h(t)$ with $|h| = o(t^{1/2})$ we have (cf. (4.16))

$$\begin{aligned} f(x + h, t\mu, tu) &= f(x + h, t\mu, tu) - f(x, 0, 0) \\ &\geq \max_{z^* \in \Omega(x)} [-t\lambda_o \mu + t\mathcal{L}_u(z^*, x, 0) \\ &\qquad + \mathcal{L}''(z^*, x, 0)((h, tu), (h, tu))] + o(|h|^2 + t^2) \\ &\geq -t\gamma(\mu - \mathbf{q}(u)) + o(t), \end{aligned}$$

so that for $\mu < \mathbf{q}(u)$ we have

$$r_n \geq p(t_n \mu, t_n u) - p(0, 0) \geq f(x_n + h_n, t_n \mu, t_n u) \geq t_n |\mu - \mathbf{q}(u)| + o(t_n),$$

which is a contradiction. □

Second-order estimates for the value function. We set

$$\mathbf{K}_u(x) = \{h: f_o'(x,0)(h,u) \le \mathbf{q}(x,u), f_i'(x,0)(h,u) \le 0, \\ i \in I(x); F'(x,0)(h,u) = 0\};$$

$$\mathbf{Q}(u) = \{x \in S: \mathbf{q}(x,u) \le \mathbf{q}(u)\};$$

$$\mathbf{Q}_\varepsilon(u) = \{x \in S: \mathbf{q}_\varepsilon(x,u) \le \mathbf{q}(u)\};$$

$$\mathbf{K}_{\varepsilon u}(x) = \{h: f_o'(x,0)(h,u) \le \mathbf{q}(x,u) + \varepsilon, f_i'(x,0)(h,u) \le \varepsilon, \\ i \in I(x), \|F'(x,0)(h,u)\| \le \varepsilon\};$$

$$\Lambda_{1u}(x) = \{z^*/\lambda_o: z^* \in \Omega_{(\mathbf{q}(u),u)}(x), \lambda_o > 0\};$$

$$\mathbf{b}(u,w) = \inf_{x \in Q(u)} \inf_{h \in \mathbf{K}_u(x)} \sup_{z^* \in \Lambda_{1u}(x)} [\mathcal{L}''(z^*,x,0)((h,u),(h,u)) + 2\mathcal{L}_u(z^*,x,0)w];$$

$$\mathbf{b}_1(u,w,\varepsilon) = \liminf_{\substack{y \to S \\ t \searrow 0}} \sup_{x \in \mathcal{M}(y,\varepsilon,t)} \sup_{z^* \in \Lambda_{1u}(x)} t^{-2}[\mathcal{L}''(z^*,x,0)((y-x,tu),(y-x,tu)) \\ + 2t^2\mathcal{L}_u(z^*,x,0)u];$$

$$\mathbf{b}_1(u,w,\varepsilon,r) = \inf_{x \in \mathbf{Q}_\varepsilon(u)} \inf_{\substack{|h| \le r \\ h \in \mathbf{K}_{\varepsilon u}(x)}} \sup_{z^* \in \Lambda_{1u}(x)} [\mathcal{L}''(z^*,x,0)((h,u),(h,u)) + 2\mathcal{L}_u(z^*,x,0)w].$$

Observe that if $\mathbf{q}(x,u) = \mathcal{L}_u(z^*,x,0)u$ for a $z^* \in \Lambda_1(x)$, then we can write equalities in the definition of $\mathbf{K}_u(x)$ for $i = 0$ and those $i \in I(x)$ for which $\lambda_i > 0$. Indeed, in this case, for $h \in \mathbf{K}_u(x)$,

$$\mathbf{q}(x,u) = \mathcal{L}_u(z^*,x,0)u = \mathcal{L}'(z^*,x,0)(h,u) \le \mathbf{q}(x,u),$$

whence the equalities.

PROPOSITION 4.8. *We have*

$$\mathbf{Q}(u) = Q(\mathbf{q}(u),u); \quad \mathbf{K}_u(x) = K_{(\mathbf{q}(u),u)}(x) \quad \text{if } x \in \mathbf{Q}(u);$$
$$\mathbf{Q}_\varepsilon(u) = Q_\varepsilon(\mathbf{q}(u),u); \quad K_{\varepsilon(\mathbf{q}(u),u)}(x) \subset \mathbf{K}_{2\varepsilon u}(x) \quad \text{if } x \in \mathbf{Q}_\varepsilon(u);$$
$$\Lambda_{1u}(x) = \arg\max_{z^* \in \Lambda_1(x)} \mathcal{L}_u(z^*,x,0)u \quad \text{if } x \in \mathbf{Q}(u).$$

Proof. According to the definitions of §3

$$Q_\varepsilon(\mu,u) = \{x \in S: q(x,\mu,u) \le q(\mu,u) + \varepsilon\},$$
$$K_{(\mu,u)}(x) = \{h: -\mu + f_o'(x,0)(h,u) \le q(x,\mu,u) + \varepsilon,$$
$$f_i'(x,0)(h,u) \le q(x,\mu,u) + \varepsilon, \quad i \in I(x), \quad \|F'(x,0)(h,u)\| \le q(x,\mu,u) + \varepsilon\}$$

for any $\varepsilon \ge 0$. We see, in particular, that $\mathbf{Q}_o(u) = \mathbf{Q}(u)$, etc. We have, furthermore (as $q(\mathbf{q}(u),u) = 0$),

$$Q_\varepsilon(\mathbf{q}(u),u) = \{x: q(x,\mathbf{q}(u),u) \le q(\mathbf{q}(u),u) + \varepsilon\} \\ = \{x: q(x,\mathbf{q}(u),u) \le \varepsilon\} = \{x: \mathbf{q}_\varepsilon(x,u) \le \mathbf{q}(u)\} = \mathbf{Q}_\varepsilon(u);$$

and for $x \in Q_\varepsilon(\mathbf{q}(u),u)$

$$K_{\varepsilon(\mathbf{q}(u),u)}(x) = \{h: -\mathbf{q}(u) + f_o'(x,0)(h,u) \le q(x,\mathbf{q}(u),u) + \varepsilon; \\ f_i'(x,0)(h,u) \le q(x,\mathbf{q}(u),u) + \varepsilon, i \in I(x), \\ \|F'(x,0)(h,u)\| \le q(x,\mathbf{q}(u),u) + \varepsilon\} \\ = \{h: f_o'(x,0)(h,u) \le \mathbf{q}(u) + 2\varepsilon; f_i'(x,0)(h,u) \le 2\varepsilon, \\ i \in I(x); \|F'(x,0)(h,u)\| \le 2\varepsilon\} \\ \begin{cases} \subset \mathbf{K}_{2\varepsilon u}(x) & \text{if } \varepsilon > 0, \\ = \mathbf{K}_u(x), & \text{if } \varepsilon = 0. \end{cases}$$

Finally, if $x \in \mathbf{Q}(u)$, then $\mathbf{q}(x,u) = \mathbf{q}(u)$ (a finite number) and hence

$$-\lambda_o \mathbf{q}(u) + \mathcal{L}_u(z^*, x, 0)u = q(\mathbf{q}(u), u) = 0$$

for any $z^* \in \Omega_{(\mathbf{q}(u),u)}(x)$. Therefore, for such x the inclusion $z^* \in \Lambda_{1u}(x)$ is equivalent to $\mathbf{q}(u) = \mathcal{L}_u(z^*, x, 0)u$. On the other hand the inequality $-\lambda_o \mu + \mathcal{L}_u(z^*, x, 0)u \le q(x, \mu, u)$ is valid for all $z^* \in \Omega(x)$ by definition. □

To obtain second-order estimates for the value function of (**P**) from the corresponding results of §3, we also need to modify the tameness condition (T) in terms of the problem. So we say that the perturbation u is *tame* if

(**T**) $$\lim_{\varepsilon \to 0} \inf_{x \in \mathbf{Q}_\varepsilon(u)} \inf\{|h|: h \in \mathbf{K}_{\varepsilon u}(x)\} < \infty.$$

THEOREM 4.9. *We assume* (UL) *and either* (**SQG**) *and* (**SN**) *or* (LG) *and* (N). *Then*

$$\mathbf{b}(u, w) \ge d_+^{(2)} v(0; u; w).$$

If in addition (**T**) *is valid, then for any* $\varepsilon > 0$,

$$d_-^{(2)} v(0; u; w) \ge \mathbf{b}^-(u, w),$$

where $\mathbf{b}^-(u, w)$ *is either* $\mathbf{b}_1(u, w, \varepsilon)$ *or* $\mathbf{b}_2(u, w, \varepsilon, r)$ (*for r being sufficiently large*).

Proof. By Theorems 2.1 (inequality 2.6_+), 4.2, 4.7, and 3.4

$$\begin{aligned} d_+^{(2)} v(0; u) &\le \inf\{\eta: p''_+(0; (\mathbf{q}(u), u), (\eta, w)) \le 0\} \\ &\le \inf\{\eta: b(0; (\mathbf{q}(u), u), (\eta, w)) \le 0\}, \end{aligned}$$

where $b(0; (\mu, u), (\eta, w))$ is equal to

$$\inf_{x \in Q(\mu,u)} \inf_{h \in K_{(\mu,u)}(x)} \max_{z^* \in \Omega_u(x)} [-2\lambda_o \eta + \mathcal{L}''(z^*, x, 0)((h, u), (h, u)) + 2\mathcal{L}_u(z^*, x, 0)w].$$

The first inequality now follows from Propositions 4.1 and 4.8.

To prove the second inequality, we first observe that, as follows from Proposition 4.8, (T) is satisfied for (C(μ, u)) if (**T**) is valid, and then, applying Theorems 2.1 (inequality (2.6_-)), 4.7, 3.13, and 3.14, and Proposition 4.1, we conclude the proof. □

Behavior of the solution map. We set (following the rules of translation under (N))

$$\mathbf{a}(x, u) = \inf_{h \in K(x)} \sup_{z^* \in \Lambda_1(x)} \left(\frac{1}{2} \mathcal{L}_{xx}(z^*, x, 0)(h, h) + \mathcal{L}_u(z^*, x, 0)u \right);$$

$$\mathbf{b}(x, u, w) = \inf_{h \in \mathbf{K}_u(x)} \sup_{z^* \in \Lambda_{1u}(x)} [\mathcal{L}_{xx}(z^*, x, 0)((h, u), (h, u)) + 2\mathcal{L}_u(z^*, x, 0)w];$$

$\mathbf{A}(x, u)$ = the set of h at which the infimum in the definition of $\mathbf{a}(x, u)$ is attained;

$\mathbf{B}(x, u, w)$ = the set of h at which the infimum in the definition of $\mathbf{b}(x, u, w)$ is attained.

THEOREM 4.10. *Assume that* $x \in S$ *is such that* $\Lambda_1(x) \ne \emptyset$ *and* $a(x, u) < \infty$. *Then*

$$D_+^{1/2} S(x, u) \subset \mathbf{A}(x, u).$$

Moreover, if $D_+^{1/2}S(x,u) \neq \emptyset$, *then* $d_+v(0;u) = \mathbf{a}(x,u)$, *and we actually have the equality*

$$D_+^{1/2}S(x,u) = \mathbf{A}(x,u).$$

Proof. Let $h \in D_+^{1/2}S(x,u)$. By Theorem 2.2, $h \in D_+^{1/2}M(x,\mu,u)$ for any finite $\mu \geq d_+v(0;u)$ and Theorem 3.2 implies that

$$h \in A(x,\mu,u), \qquad p'_+(0;(\mu,u)) = a(x,\mu,u),$$

where, as usual,

$$a(x,\mu,u) = \inf_{h\in K(x)} \max_{z^*\in\Omega(x)} \left(-\lambda_o\mu + \frac{1}{2}\mathcal{L}_{xx}(z^*,x)(h,h) + \mathcal{L}_u(z^*,x,0)u\right)$$

and $A(x,\mu,u)$ is the collection of h at which the infimum in the definition of $a(x,\mu,u)$ is attained.

Taking into account that $a(x,\mu,u) \geq a(\mu,u)$ and applying Theorem 2.1, we conclude that

$$d_+v(0;u) = \inf\{\mu\colon a(x,\mu,u) \leq 0\}.$$

As $\Lambda_1(x) \neq \emptyset$, it follows (also from Proposition 4.1) that $d_+v(0;u)$ is finite and equal to $\mathbf{a}(x,u)$ and $A(x,d_+(0;u),u) = \mathbf{A}(x,u)$.

Furthermore, if $h \in A(x,u)$, Theorem 3.2 implies the existence of a curve $h(t) \to h$ such that

$$f(x+\sqrt{t}h(t), td_+v(0;u), tu) \leq p(tdv_+(0;u), tu) + o(t).$$

In view of the obvious inequality

$$p(t\mu, tu) \leq t[p'_+(0;(\mu,u)) + r(t)],$$

where $r(t) \to 0$ as $t \to 0$, and the fact that $p'_+(0;(dv_+(0;u),u)) \leq 0$ by Theorem 2.1, this means that for any $\varepsilon > 0$

$$\begin{aligned}
&f_i(x+\sqrt{t}h(t), tu) \leq tr(t) \leq t\varepsilon, \qquad i = 1,\dots,m;\\
&\|F(x+\sqrt{t}h(t), tu)\| \leq tr(t) \leq t\varepsilon;\\
&f_o(x+\sqrt{t}h(t), tu) \leq \theta + t[d_+v(0;u) + r(t)].
\end{aligned}$$

The first two inequalities show that $x+\sqrt{t}h(t)$ is feasible in $(\mathbf{P}_{\varepsilon(t)}(tu))$, whereas the third implies that

$$v_{t\varepsilon}(tu) \leq f_o(x+\sqrt{t}h(t), tu) \leq \theta + t[d_+v(0;u) + r(t)].$$

Let the sequence of t_n be such that

$$t_n^{-1}(v_{t_n\varepsilon}(t_nu) - \theta) \to d_+v(0;u).$$

Then for any $\delta > 0, x+\sqrt{t_n}h(t_n)$ is $t_n\delta$-optimal in $(\mathbf{P}_{\varepsilon(t_n)}(t_nu))$ if n is sufficiently large, that is,

$$x+\sqrt{t_n}h_n \in S_{t_n\varepsilon, t_n\delta}(t_nu),$$

and this implies that $h \in D_+^{1/2}S(x,u)$. □

THEOREM 4.11. *Assume that* $x \in S$ *is such that* $|\mathbf{q}(x,u)| < \infty$, $\Lambda_{1u}(x) \neq \emptyset$, *and* $|\mathbf{b}(x,u,w)| < \infty$. *Then*

$$D_+S(x,u,w) \subset \mathbf{B}(x,u,w).$$

Moreover, if $\mathbf{D}_+(x,u,w) \neq \emptyset$, *then* $d_+v(0;u) = \mathbf{q}(u) = \mathbf{q}(x,u)$, $d_+^{(2)}v(0,u,w) = \mathbf{b}(x,u,w)$, *and, if in addition* $dv(0;u)$ *exists, then*

$$D_+S(x,u,w) = \mathbf{B}(x,u,w).$$

Proof. We have $\mathbf{a}(x,u) \leq \mathbf{q}(x,u) < \infty$. So if $h \in D_+S(x,u,w)$, then $0 \in D_+^{1/2}S(x,u)$ and by Theorem 4.11 $0 \in \mathbf{A}(x,u)$. Hence, $\mathbf{a}(x,u) = \mathbf{q}(x,u)$, $d_+v(0;u) = \mathbf{q}(x,u)$, and consequently, $\mathbf{q}(u) = \mathbf{q}(x,u)$ by Theorem 4.2. In particular, $d_+v(0;u)$ is finite and we can apply Theorem 2.2 to conclude that $h \in D_+M(x,(d_+v(0;u),u),(\eta,w))$ for any finite $\eta \geq d_+^{(2)}v(0,u,w)$.

The rest of the proof is similar to that of the preceding theorem. From Theorem 3.4 we deduce that for any such η

$$h \in B(x,(d_+v(0;u),u),(\eta,w)) \quad \text{and}$$
$$p''_+(0;(d_+v(0;u),u),(\eta,w)) = b(x,(d_+v(0;u),u),(\eta,w)),$$

where $b(x,(\mu,u),u),(\eta,w)$ is equal to

$$\inf_{h\in K_{(\mu,u)}(x)} \max_{z^*\in\Omega_{(\mu,u)}(x)} [-2\lambda_o\eta + \mathcal{L}''(z^*,x,0)((h,u),(h,u)) + 2\mathcal{L}_u(z^*,x,0)w]$$

and $B(x,(\mu,u),u),(\eta,w)$ is the collection of h at which the infimum above is attained, and a routine application of Proposition 4.1 (possible because $\Lambda_{1u}(x) \neq \emptyset$) and Theorem 2.1 gives the equality $d_+^{(2)}v(0,u,w) = \mathbf{b}(x,u,w)$ and the inclusion for $\mathbf{D}_+S(x,u,w)$.

If we assume that $dv(0;u)$ exists, then for any given $h \in D_+M(x,(dv(0;u),u),(\eta,w))$ we can find a trajectory $h(t) \to h$ such that for $x(t) = x + th(t), \mu(t) = tdv(0;u) + (t^2\backslash 2)d_+^{(2)}v(0,u,w)$, and $u(t) = tu + t^2w$, we have

$$0 \leq f(x(t),\mu(t),u(t)) \leq p(\mu(t),u(t)) + t^2r(t),$$

where $r(t) \to 0$ as $t \to 0$, from which, as in the proof of Theorem 4.11, we find that $x \in S_{\varepsilon(t_n),\delta(t_n)}(u_n)$, where $\varepsilon(t) \leq \varepsilon t^2$ and $\delta(t) \leq \delta t^2$, where $\{t_n\}$ is a sequence along which $d_+^{(2)}v(0,u,w)$ is realized. □

We conclude this subsection by proving three stability results for approximate solutions of $(\mathbf{P}(u))$ that are simple corollaries of the corresponding facts established in §3.

Let us say for brevity that the perturbation u is *first-order feasible* if $-\infty < d_-v(0;u) \leq d_+v(0;u) < \infty$. By Propositions 4.4 and 4.5 a sufficient condition for u to be first-order feasible is that (4.12) holds and the normality condition (N) is satisfied.

THEOREM 4.12. *We assume that* u *is first-order feasible and the conditions* (QG) *and* (UL) *hold. Then for any* $\eta > 0$ *and sufficiently small* $\varepsilon > 0$

$$\limsup_{t\searrow 0} t^{-1}\sigma^2(S_{t\varepsilon,t\eta}(tu),S) < \infty. \tag{4.17}$$

Proof. By Theorem 2.2, $M = S$ and

$$\sigma(S_{\varepsilon,\delta}(u),S) \leq \sigma(M_{\varepsilon+\delta}(\mathbf{p}(u,\varepsilon) - \theta,u),M). \tag{4.18}$$

Since u is first-order feasible, there is an $N > 0$ such that

$$|v_{t\varepsilon}(tu) - \theta| \leq Nt$$

if ε and t are sufficiently small. Applying Theorem 3.5 to $f(x, \mu, u)$ with $\mu(t) = t^{-1}(v_{t\varepsilon}(tu) - \theta)$, we conclude that for small ε and any η

$$\limsup_{t \searrow 0} t^{-1}\sigma^2(M_{\eta t}(v_{t\varepsilon}(tu) - \theta, tu), M) < \infty. \qquad \square$$

In a similar manner we say that the pair (u, w) is *second-order feasible* if $dv(0; u)$ exists and is finite and $-\infty < d_-^{(2)}v(0; u; w), d_+^{(2)}v(0, u; w) < \infty$.

THEOREM 4.13. *We assume that the pair (u, w) is second-order feasible, and the conditions* (**SQG**), (**T**), *and* (UL) *hold. Then*

$$\limsup_{t \searrow 0} t^{-1}\sigma(S_{\varepsilon(t),\eta(t)}(tu + t^2 w), S) < \infty, \tag{4.19}$$

provided $\eta(t) = O(t^2), \varepsilon(t) = \varepsilon t^2$, and ε is sufficiently small.

Proof. By definition, (SQG) holds for $f(x, \mu, u)$ along $(\mathbf{q}(u), u)$ and by Theorem 4.7 $dv(0; u) = \mathbf{q}(u)$. Since (u, w) is second-order feasible,

$$v_{\varepsilon(t)}(tu + t^2 w) = \theta + t \cdot d\mathbf{p}(0; u) + \eta(t, \varepsilon)t^2, \quad \text{where } \limsup_{\varepsilon, t \searrow 0} |\eta(t, \varepsilon)| < \infty.$$

Therefore, by Theorem 3.11,

$$\limsup_{t \searrow 0} t^{-1}\sigma(M_{\varepsilon(t),\zeta(t)}(v(tu + t^2 w) - \theta, tu + t^2 w), M) < \infty$$

provided $\zeta(t) = O(t^2)$, and, again, application of (4.18) completes the proof. $\square$

The same argument as in the proof of Theorem 4.12 but with the reference to Theorem 3.16 instead of 3.5 gives the following result.

THEOREM 4.14. *We assume that u is first-order feasible and* (LG) *and* (UL) *are valid. Then for any $\eta > 0$ and sufficiently small $\varepsilon > 0$*

$$\limsup_{t \searrow 0} t^{-1}\sigma(S_{t\varepsilon,t\eta}(tu), S) < \infty.$$

Special cases. The case of a finite solution set. In this case the existence of the first and second variational derivatives of the value function is guaranteed under mild additional assumptions.

THEOREM 4.15. *Assume that S is a finite set and for any $x \in S$ the set $\Lambda_1(x)$ is nonempty and the image of the linear map $h \to F_x(x, 0)h$ is closed. If under these assumptions*

(a) (QG) *is valid, then $dv(0; u)$ exists for any u and*

$$dv(0; u) = \mathbf{a}(u);$$

(b) (**SQG**) *is valid, $|\mathbf{q}(u)| < \infty$, and $\Lambda_{1u}(x) \neq \emptyset$ for any $x \in Q(u)$, then both $dv(0; u)$ and $d^{(2)}v(0; u, w)$ exist and*

$$dv(0; u) = \mathbf{q}(u), \qquad d^{(2)}v(0; u, w) = \mathbf{b}(u, w).$$

Proof. (a) By Theorem 2.1 we must show that $p(\mu, u)$ is differentiable at zero along every direction and that $p'(0; (\mu, u)) = a(\mu, u)$.

To this end we have to verify, as in the proof of Theorem 3.10, that in our case, when

$$\Phi(x,\mu,u) = (f_o(x,u) - \mu - \theta, f_1(x,u), \dots, f_m(x,u), F(x,u)),$$
$$g(z) = \max\{\alpha_o, \dots, \alpha_m, \|y\|\},$$

the inequality

$$\|K(x) - h\| \le c \cdot g'(\Phi(x); \Phi_x(x,0)h)$$

is valid for all $x \in S = M$ and all h if $c > 0$ is sufficiently large.

The latter reduces to

$$\|K(x) - h\| \le c \cdot \max\{\max_{i \in I(x)} f_{ix}(x,0)h, \|F_x(x,0)h\|\}$$

and this follows from the well-known Hoffmann's inequality valid if the image of $F_x(x,0)$ is closed in Y (as in [33]).

(b) To prove the second part, we first observe that $\mathbf{Q}(u) \neq \emptyset$ and $\mathbf{Q}_\varepsilon(u) = \mathbf{Q}(u)$ for sufficiently small ε (which follows from that facts that S is a finite set and $\mathbf{q}(u)$ is finite). For the same reason, (**SN**) follows from the fact that $\Lambda_1(x) \neq \emptyset$ for any $x \in S$. The existence of $dv(0;u)$ and the equality $dv(0;u) = \mathbf{q}(u)$ are now ensured by Theorem 4.7.

The existence of and the formula for $d^{(2)}v(0;u,w)$ will be proved if we show that there is a $c > 0$ such that for all $x \in \mathbf{Q}(u)$ we have

$$\|K_u(x) - h\| \le c \cdot \max\{f_o'(x,u)(h,u) - \mathbf{q}(u), \max_{i \in I(x)\setminus\{0\}} f_i'(x,u)(h,u), \|F'(x,u)(h,u)\|\},$$

in which case the result follows from Theorem 4.9.

To prove the inequality, we consider the set

$$\mathbf{N} = \{(h,t)\colon f_o'(x,u)(h,tu) - \mathbf{q}(u) \le 0, \max_{i \in I(x)\setminus\{0\}} f_i'(x,u)(h,tu) \le 0, \|F'(x,u)(h,tu)\|\},$$

where we set $\mathbf{q}(-u) = -\mathbf{q}(u)$. It is clear that the image of the map $(h,t) \to F'(x,0)(h,tu)$ is closed. Therefore, as in the proof of the first part of the theorem, we conclude that the inequality

$$\|\mathbf{N} - (h,t)\| \le c \cdot \max\{f_o'(x,u)(h,u) - \mathbf{q}(u), \max_{i \in I(x)\setminus\{0\}} f_i'(x,u)(h,u), \|F'(x,u)(h,u)\|\}$$

is valid for all (h,t). Taking $t = 1$, we obtain the desired result.

Regular problems. Let us now consider what additional information we may get in the presence of constraint qualification. Specifically, we shall discuss two types: the Mangasarian–Fromowitz constraint qualification and the directional regularity condition of the Gollan type.

Recall the Mangasarian–Fromowitz condition for (**P**):

(MF) Im $F_x(x,0) = Y$ and there is an $h \neq 0$ such that

$$F_x(x,0)h = 0; \quad f_{ix}(x,0)h < 0, \quad i \in I\setminus\{0\}.$$

Take a sufficiently large $N > 0$ and consider the function

$$\psi_N(x,u) = f_o(x,u) + N\left(\sum_{i=1}^m f_i^+(x,u) + \|F(x,u)\|\right),$$

where $\alpha^+ = \max\{\alpha, 0\}$. For this function we consider the problem of unconstrained minimization over $x \in D$ and denote by $p_\psi(u), M_\psi$, and $\Omega_\psi(x)$ the value function, the set of solutions of the unperturbed problem, and the collection of Lagrange multipliers at $x \in M_\psi$.

THEOREM 4.16. *Assume that S is a compact set and* (MF) *is valid for all $x \in S$. Then there are $N > 0$ and $\gamma > 0$ such that for any neighborhood D' of S with $\sigma(D', S) < \gamma$ the problem* (**P**(u)) *with D replaced by D' and $\|u\| < \gamma$ and the problem of unconstrained minimization of ψ_N over $x \in D'$ are equivalent in the sense that*

(a) $v(u) = p_\psi(u)$ *and* $S = M_\psi$;

(b) $\Omega_\psi(x) = \Lambda(x)$.

In particular, the value function $v(u)$ is Lipschitz continuous at zero.

Proof. The (MF)-condition at x implies that $\Lambda(x)$ is bounded. Simple compactness arguments show that the union of $\Lambda(x)$ over $x \in S$ is also bounded and the observation that

$$\Omega_\psi(x) = \{z^* = (1, \lambda_1, \dots, \lambda_m, y^*) \colon \lambda_i \geq 0, \lambda_i f_i(x) = 0, i = 1, \dots, m; \\ 0 \leq \lambda_i \leq N, i = 1, \dots, m, \mathcal{L}_x(z^*, x, 0) = 0\}$$

completes the proof of (b).

To prove (a) we apply the regularity theorem of Robinson [56]: for any $x \in S$ there are $\gamma(x) > 0, N(x) > 0$ such that the distance from w to the set of admissible elements of (**P**(u)) does not exceed

$$R(x)\left(\sum_{i=1}^m f_i^+(w,u) + \|F(w,u)\|\right),$$

provided that $\|u\| \leq \gamma(x), \|w - x\| \leq \gamma(x)$, and the (MF)-condition is satisfied at x. Again, compactness of S implies that the infimum of $\gamma(x)$ over S is positive and the supremum of $R(x)$ over S is finite.

Taking γ smaller than the first and R bigger than the second we see that for any u with $\|u\| \leq \gamma$ and any w within γ of S there is an element z that is admissible in (**P**(u)) and lies within

$$R\left(\sum_{i=1}^m f_i^+(w,u) + \|F(w,u)\|\right).$$

of w. Therefore, if L is the Lipschitz constant of f_o in the γ-neighborhood of S, then setting $N = LR$, we conclude that

$$v(u) \leq f_o(z,u) \leq f_o(w,u) + L\|w - z\| \leq \psi_N(w,u).$$

On the other hand, if x is a solution of (**P**(u)), then obviously $\psi_N(x,u) = F_o(u)$. This completes the proof of the theorem.

We observe now that (PP) is satisfied, so all the results of §3 apply to this function. We further notice that the Lagrangian functions of the two problems are also identical, so applying the results of §3 together with Theorem 4.16, we arrive at the same sensitivity estimates but variational derivatives of $v(\cdot)$ replaced by the usual Dini directional derivatives as in §3.

The latter is true under a weaker regularity condition extending to an infinite-dimensional setting the directional regularity condition introduced by Gollan [29] for problems in a finite-dimensional space.

(DR) Im $F_x(x,0)$ *is a closed subspace of* Y *and*

$$\mathcal{L}_u(z^*,x,0)u < 0, \quad \forall\, z^* \in \Lambda_o(x), \quad z^* \neq 0.$$

THEOREM 4.17. *We assume that* S *is a compact set and the following condition (the compactness assumption) holds: if* $t \to 0$ *and* $f_o(x,tu) \to \theta, f(x,tu) \to 0$, $F(x,tu) \to 0$, *then* $\|S - x\| \to 0$. *Assume further that, for a given* u, (DR) *is valid for every* $x \in S$. *Then*

$$d_\pm v(0;u) = v'_\pm(0;u)$$

and

$$d^{(2)}_\pm v(0;u,w) = v''_\pm(0;u,w), \quad \forall\, w.$$

Here $v'_\pm(0;u)$ and $v''_\pm(0;u)$ are the upper and lower Dini directional derivatives of $v(\cdot)$ of the first and second order (as were defined for $p(\cdot)$ in §2).

Proof. First, we notice that simple arguments (cf. [3]) show that (DR) is the same as

$$\text{Im } F_x(x,0) = Y$$

and there is an $h \in X$ such that

$$F_x(x,0)h + F_u(x,0)u = 0, \quad f_{ix}(x,0)h + f_{iu}(x,0)u < 0, \quad i \in I(x)\backslash\{0\}. \tag{4.20}$$

This, again, allows us to apply the Robinson regularity theorem to the system

$$\mathbf{F}(\mathbf{x}) = 0, \quad \mathbf{f}_i(\mathbf{x}) \leq 0, \quad i = 1,\dots,m,$$

where $\mathbf{x} = (x,t), \mathbf{F}(\mathbf{x}) = F(x,tu)$, $\mathbf{f}_i(\mathbf{x}) = f_i(x,tu)$, so that we can find an $N > 0$ and $\gamma > 0$ such that, whenever the distance from x to S is smaller than γ and $|t| < \gamma$, there are x' and t' such that x' is admissible in $(\mathbf{P}(t'u))$ and

$$\|x - x'\| + |t - t'| \leq N\left(\sum f_i^+(x,tu) + \|F(x,tu)\|\right).$$

Now if $x \in S_{t\varepsilon,t\delta}(tu)$, that is, $\|F(x,tu)\| \leq t\varepsilon$, $f_i(x,tu) \leq t\varepsilon$, and $f_o(x,tu) \leq v_{t\varepsilon}(tu) + t\delta$, then $\|x - x'\|, |t - t'| \leq tN(m+1)\varepsilon$, and, if L is a Lipschitz constant of f_o in a neighborhood of $S \times \{0\}$,

$$\begin{aligned} v(t'u) \leq f_o(x',t'u) &\leq f_o(x,tu) + tLN(m+1)\varepsilon \\ &\leq v_{t\varepsilon}(tu) + +t(LN(m+1)\varepsilon + \delta). \end{aligned}$$

This immediately implies the desired equalities for the lower derivatives, for as $t'/t \to 1$, we have

$$v'_-(0;u) \leq \liminf_{t\to 0} t^{-1}v(t'u) \leq d_-v(0;u)$$

and the opposite inequality follows from the definitions. The same, of course, is true for the second lower derivatives.

The proof for the upper derivatives is more complicated. Essentially, what we have to show is the possibility of replacing t' by t, at least for those t that determine the upper Dini directional derivatives of $v(\cdot)$. We shall do this in several steps.

(1) First, there are $N > 0$ and $\gamma > 0$ (maybe different from those above) such that

(a) for any x, t with $\|S - x\| < \gamma$, $0 \le t < \gamma$ there is an x' such that

$$F(x', tu) = 0, \qquad \|x' - x\| \le N\|F(x, tu)\|;$$

(b) for any $h \in \mathcal{X}$, any $y \in \mathcal{Y}$, and any $x \in S$ there is h' such that

$$F_x(x, 0)h' = y, \quad \|h' - h\| \le N\|F_x(x, 0)h - y\|.$$

This follows from the theorem of Ljusternik–Graves (see [37], for example), or we can refer again to the Robinson's regularity theorem; (b) in fact follows from the Banach open mapping theorem.

From (a), (b) we conclude that *there are* $K > 0$, $\alpha > 0$ *such that for any* $x \in S$ *there is an* $h = h(x)$ *satisfying*

(4.21)
$$\|h\| \le K; \quad F_x(x, 0)h + F_u(x, 0)u = 0, \quad f_{ix}(x, 0)h + f_{iu}(x, 0)u < -\alpha, \quad i \in I(x)\backslash\{0\}.$$

Indeed, fix an $z \in S$ and take an $h = h(z)$ satisfying (4.20). Denote $K(z) = 2\|h\|$ and $2\alpha(z) = -\max\{f_{ix}(z, 0)h + f_{iu}(z, 0)u : i \in I(z)\backslash\{0\}\}$.

Now choose a positive $\delta' \le K(z)$ such that

$$f_{ix}(x, 0)h' + f_{iu}(x, 0)u \le -\alpha(z), \quad \forall\, i \in I(x)\backslash\{0\} \tag{4.22}$$

whenever $\|z - x\| < \delta'$, $\|h - h'\| < \delta'$, and take a positive $\delta(z) = \delta < \delta'$ such that $N\|F_x(x, 0)h + F_u(x, 0)u\| < \delta'$ if $\|x - z\| < \delta$. Then, given such an x, we find, thanks to (b), an $h' = h(x)$ such that

$$F_x(x, 0)h' + F_u(x, 0)u = 0, \qquad \|h' - h\| < \delta'.$$

In other words, (4.21) is valid for all x of the δ-neighborhood of z. The rest follows from compactness of S.

Set

$$C = \sup_{\substack{\operatorname{dist}(S, x) < \gamma \\ 0 \le t < \gamma}} \max\{\|F_x(x, tu)\|, \max_i \|f_{ix}(x, tu)\|\}.$$

In the argument to follow we fix these constants: α, γ, C, K, N.

(2) Now let the sequence of $t_n \to 0$ be such that

$$t_n^{-1}(v(t_n u) - \theta) \to v'_+(0; u).$$

Fix a small $\varepsilon > 0$, take a $\xi \in (0, 1)$, to be specified later, and consider the problem $(\mathbf{P}_{\tau_n \varepsilon}(\tau_n u))$, where $\tau_n = (1 - \xi)t_n$. Let x_n be an $o(\tau_n)$-solution to the problem, that is,

$$\limsup_{n \to \infty} \tau_n^{-1}(f_o(x_n, \tau_n u) - \theta) = \limsup_{n \to \infty} \tau_n^{-1}(v_{\tau_n \varepsilon}(\tau_n u) - \theta). \tag{4.23}$$

Then by the compactness assumption, $\|S - x_n\| \to 0$, and we can assume without loss of generality that x_n converges to a certain $x \in S$. Take an $h = h(x)$ satisfying (4.21) and choose an $\eta > 0$ to ensure that for any z, t satisfying $\|z - x\| < \eta$, $0 \le t < \eta$

(i) $\eta < \gamma$;

(ii) $f_i(z, tu) < 0$ if $i \notin I(x)\backslash\{0\}$;

(iii) $\|F_x(z, tu)h + F_u(z, tu)u\| < \varepsilon$; $f_{ix}(x, tu)h' + f_{iu}(x, tu)u \le -\alpha/2$, which is possible thanks to (4.21).

Set

$$x'_n = x_n + (t_n - \tau_n)h = x_n + \xi t_n h,$$

and let z_n satisfy the relations (cf. (a) in (1))

$$F(z_n, t_n u) = 0; \qquad \|z_n - x'_n\| \le N\|F(x'_n, t_n u)\|.$$

We have

$$F(x'_n, t_n u) = F(x_n, t_n u) + \int_0^1 [F_x(x_n(\sigma), \tau_n(\sigma)u)\xi t_n h + F_u(x_n(\sigma), \tau_n(\sigma)u)\xi t_n u]d\sigma,$$

where $x_n(\sigma) = x_n + \sigma\xi t_n h$, $\tau_n(\sigma) = \tau_n + \sigma\xi t_n$, so that (by (iii))

$$\|F(x'_n, t_n u)\| \le \|F(x_n, t_n u)\| + \xi t_n \varepsilon \le \tau_n \varepsilon + \xi t_n \varepsilon = t_n \varepsilon.$$

Likewise, for $i \in I(x)\backslash\{0\}$,

$$\begin{aligned} f_i(x'_n, t_n u) &= f_i(x_n, t_n u) + \xi t_n \int_0^1 [f_{ix}(x_n(\sigma), \tau_n(\sigma)u)h + f_{iu}(x_n(\sigma), \tau_n(\sigma)u)u]d\sigma, \\ &\le \tau_n \varepsilon - \xi t_n(\alpha/2) = t_n((1-\xi)\varepsilon - \xi(\alpha/2)), \end{aligned}$$

and consequently, for such i,

$$\begin{aligned} f_i(z_n, t_n u) &\le t_n((1-\xi)\varepsilon - \xi(\alpha/2)) + C\|z_n - x'_n\| \\ &\le t_n[NC\varepsilon + (1-\xi)\varepsilon - \xi(\alpha/2)] \le 0 \end{aligned}$$

if, say,

$$\xi = \frac{1 + CN}{1 + \alpha/2}\varepsilon = p\varepsilon,$$

where p is an absolute constant.

On the other hand, for $i \notin I(x)$, we have by (ii) that $f_i(z_n, t_n u) < 0$ if n is sufficiently large. This means that z_n is admissible in $(\mathbf{P}(t_n u))$.

We have (for $\xi = p\varepsilon$)

$$\|z_n - x_n\| \le Nt_n\varepsilon + Kpt_n\varepsilon$$

so that

$$\begin{aligned} f_o(z_n, t_n u) &\le f_o(x'_n, \tau_n u) + R(\|z_n - x'_n\| + |t_n - \tau_n|) \\ &\le f_o(x'_n, \tau_n u) + R(N + (K+1)p)t_n\varepsilon, \end{aligned}$$

where

$$R = \sup_{\substack{\|S-x\|<\gamma \\ 0\le t<t}} \max\{\|f_{0x}(x, tu)\|, \|f_{0u}(x, tu)\| \cdot \|u\|\}.$$

It follows from (4.23) that

$$\begin{aligned} v'_+(0; u) &\le \limsup_{n\to\infty} t_n^{-1}(f_o(z_n, t_n u) - \theta) \\ &\le \limsup_{n\to\infty} t_n^{-1}(f_o(x'_n, \tau_n u) - \theta) + R(N + (K+1)p)t_n\varepsilon \\ &\le (1 - p\varepsilon) \limsup_{n\to\infty} \tau_n^{-1}(v_{\tau_n\varepsilon}(\tau_n u) - \theta) + R(N + (K+1)p)t_n\varepsilon. \end{aligned}$$

As this is true for any $\varepsilon > 0$, we conclude that

$$v'_+(0; u) \le d_+ v(0; u).$$

Similar arguments prove the corresponding inequality for the second derivatives.

Linear programming. The last application we consider will be to the standard problem of linear programming with the data depending on a parameter u:

$$\textbf{(LP}(u)\textbf{)} \qquad \begin{aligned} &\text{minimize } c^{\mathrm{T}}(u)x \\ &\text{subject to } A(u)x \geq b(u), \qquad z \geq 0 \end{aligned}$$

with $x \in X = R^n$, $A(u)\colon R^n \to R^m$, and $u \in R^k$.

As always, we assume that $u = 0$ is the nominal value of the parameter, and that S, the set of solutions of the unperturbed problem, is nonempty. Then $\Lambda_1(x) = S^*$ for all x, where S^* is the set of solutions of the dual problem

$$\begin{aligned} &\text{maximize } b^{\mathrm{T}}(u)y \\ &\text{subject to } A^{\mathrm{T}}(u)y \leq c(u), \qquad y \geq 0, \end{aligned}$$

corresponding to the nominal value of the parameter.

For the Lagrangian of **(LP)** we have the expression

$$\mathcal{L}(y, x, u) = c^{\mathrm{T}}(u)x + y^{\mathrm{T}}(b(u) - A(u)x).$$

THEOREM 4.18. **(LP)** *satisfies* (LG) *and if S is bounded, then it also satisfies* (UL). *Thus, in this case $dv(0; u)$ exists for any u and*

$$dv(0; u) = \inf_{x \in S} \sup_{y \in S^*} \mathcal{L}_u(y, x, 0)u.$$

Proof. Let, as always, $\theta = v(0)$. Then S is the solution of the following system of linear inequalities:

$$Ax \geq b, \quad x \geq 0, \quad c^{\mathrm{T}}x \leq \theta,$$

and (by the already mentioned inequality of Hoffmann) there is an $N > 0$ such that

$$\begin{aligned} \|S - x\| &\leq N\left((c^{\mathrm{T}}x - \theta)^+ + \sum (b - Ax)_i^+ + \sum x_j^-\right) \\ &\leq N(n + m + 1) \cdot \max\{(c^{\mathrm{T}}x - \theta)^+, \max_i (b - Ax)_i^+, \max_j x_j^-\}. \end{aligned}$$

We observe now that

$$\begin{aligned} &\max\{(c^{\mathrm{T}}x - \theta)^+, \max_i (b - Ax)_i^+, \max_j x_j^-\} \\ &\qquad = \max\{(c^{\mathrm{T}}x - \theta), \max_i (b - Ax)_i, \max_j x_j\} = f(x), \end{aligned}$$

because $f(x) \geq 0$ for all x. This proves that (LG) holds. The proof that (UL) is valid if S is bounded is straightforward and the final statement follows from Theorem 4.2(c).

The expression for $dv(0; u)$ becomes especially symmetric if the dependence of $c(u), b(u)$, and $A(u)$ on u is affine:

$$c(u) = c + c_o(u), \quad b(u) = b + b_o(u), \quad A(u) = A + A_o(u),$$

where $c_o(u), b_o(u)$, and $A_o(u)$ are linear functions of u.

Set $\mathcal{L}_o(y, x, u) = c_o^T(u)x + y^T(b_o(u) - A_o(u)x)$. Then

$$\mathcal{L}_u(y, x, 0)u = \mathcal{L}_o(y, x, u),$$

and therefore the following is true.

THEOREM 4.19. *Assume, in addition to the condition of Theorem* 4.18, *that the dependence of* $c(u), b(u)$, *and* $A(u)$ *on* u *is affine. Then*

$$dv(0;u) = \inf_{x\in S}\sup_{y\in S^*}\mathcal{L}_o(y,x,u).$$

To conclude, we shall analyze the second-order behavior of (**LP**) in the simplest situation of Theorem 4.19.

To this end we first observe that, as $y^{\mathrm{T}}(Ax-b)=0$ for all $x\in S, y\in S$, and $c^{\mathrm{T}}x=\theta$ for all $x\in S$, we may write the equality

$$\mathcal{L}(y,x,u)-\theta=\mathcal{L}_o(y,x,u),$$

which means that the collections of saddle points of $\mathcal{L}(\cdot,\cdot,u)$ and $\mathcal{L}_o(\cdot,\cdot,u)$ on $S\times S^*$ coincide. By polyhedrality, this collection is nonempty (if, of course, $dv(0;u)$ is finite) and actually having the form of a product $S(u)\times S^*(u)$, where $S(u)\subset S$ and $S^*(u)\subset S^*$.

It is an easy matter to see that $S(u)=\mathbf{Q}(u)$ so that for any $x\in S(u)$ we have $\mathbf{q}(x,u)=\mathbf{q}(u)$.

We have, furthermore,

$$K_u(x)=\{h\colon c^{\mathrm{T}}h+c_o^{\mathrm{T}}(u)x\le\mathbf{q}(x,u);h_j\ge 0,j\in B(x);\\ (A_jh+A_o(u)x)_i\ge b_{oi}(u),i\in I(x)\}$$

(where $I(x)=\{i\colon (Ax)_i=b_i\}$ is the set of active constraints and $B(x)=\{j\colon x_j=0\}$ is the "basis" of x);

$$K_{u\varepsilon}(x)=\{h\colon c^{\mathrm{T}}h+c_o^{\mathrm{T}}(u)x\le\mathbf{q}(x,u)+\varepsilon;h_j\ge-\varepsilon,j\in B(x);\\ (A_jh+A_o(u)x)_i\ge b_{oi}(u)-\varepsilon,i\in I(x)\}.$$

Set

$$\mathcal{K}(u)=\{(x,h)\colon x\in S(u),h\in K_u(x)\},\\ \mathcal{K}_{\varepsilon r}(u)=\{(x,h)\colon x\in S_\varepsilon(u),h\in K_{u\varepsilon}(x),\|h\|\le r\},$$

where

$$S_\varepsilon(u)=\{x\in S\colon\mathbf{q}_\varepsilon(x,u)\le\mathbf{q}(u)\}.$$

Then

$$\mathbf{b}(u,w)=\inf_{(x,h)\in\mathcal{K}(u)}\sup_{y\in S^*(u)}(\mathcal{L}_o(y,h,u)+2\mathcal{L}_o(y,x,w));\\ \mathbf{b}_1(u,w,\varepsilon,r)=\inf_{(x,h)\in\mathcal{K}_{\varepsilon r}(u)}\sup_{y\in S^*(u)}(\mathcal{L}_o(y,h,u)+2\mathcal{L}_o(y,x,w)).$$

Unlike in the general case, $S^*=\Lambda_1$ does not depend on x so that $\mathbf{q}(x,u)$ is a polyhedral function of x and $\mathbf{q}_\varepsilon(x,u)$ is a polyhedral function of (x,ε). Therefore, by well-known properties of polyhedral sets and functions [59] we have in this case

$$\bigcap_\varepsilon S_\varepsilon(u)=S(u)\quad\text{and}\quad\bigcap_\varepsilon\mathcal{K}_{\varepsilon r}(u)\subset\mathcal{K}(u),$$

and applying Theorem 4.9 we arrive at the following result (observe that (**T**) holds automatically as $S \neq \emptyset$ and $K_u(x) \neq \emptyset$ for $x \in S$—the latter easily follows from duality arguments as in [3]).

THEOREM 4.20. *We posit the assumptions of Theorem* 4.19 *and assume in addition that* u *is first-order feasible. Then* $d^{(2)}(0; u, w)$ *exists for any* w *and*

$$d^{(2)}v(0; u, w) = \inf_{(x,h)\in K(u)} \sup_{y\in S^*(u)} (\mathcal{L}_o(y, h, u) + 2\mathcal{L}_o(y, x, w)).$$

5. Concluding remarks.

5.1. In the course of developments of sensitivity analysis, the regularity assumptions evolved from linear independence of all active constraints plus strict complimentarity [4], [20], [55] with the strict complementarity condition subsequently dropped [38], [57] and the linear independence condition replaced first by the (MF)-constrained qualification [43], [60], [66], [67] and further by the Gollan directional regularity condition [3], [5], [26], [29]. (For infinite-dimensional problems, a similar evolution can be partly traced, e.g., through [13], [37], [48], [71], and this article as regards the (DR)-condition.) We refer to [20], [22], and [30] for a survey of results and developments.

The linear independence conditions are not really connected with the problems studied in this article. The condition is crucial whenever we need the implicit function theorem, for example, in the "global" singularity theory in nonlinear programming as developed in [30], [39]–[41], [54], and [70].

It seems that the same must now be said about constraint qualifications: the (MF)-condition is fundamental for analytic stability properties (see [31], [43], [44], and also Remark 4 below), but not for sensitivity estimates. What is really needed is *normality* of one or another sort.

5.2. In the absence of the linear independence of gradients of active constraints, Lipschitz stability of exact solutions was first established by Robinson [58] under the (MF)-condition and a fairly strong version of second-order sufficient condition. This result was strengthened by Gauvin and Janin [26], who replaced (MF) by (DR), and Shapiro [66], who offered a weaker version of the strong second-order condition (in fact, equivalent to (**SQG**) in the case of an isolated solution with the (MF)-condition). The last two results were unified by Auslender and Cominetti [3].

5.3. It is a surprising observation that follows from Theorems 3.2, 3.4, 4.10, and 4.11 that to obtain the exact expression for the upper derivative of the solution maps we do not need any additional assumptions (second order, growth, etc.). The latter are needed only to guarantee that the corresponding upper derivative is nonempty. For that we have to verify that there are trajectories of approximate solutions with one or another rate of convergence to the solution set of the unperturbed problem. The existence of such trajectories may be itself used as an assumption instead of growth conditions or second-order sufficient conditions (cf. [6]).

It is also to be observed that the approximate solutions must be incorporated in the results of such sort: for exact solutions they may fail to be valid (see an example in [6]).

5.4. Lempio and Maurer [45] observed that (under the MF-condition) the Lipschitz upper continuity of approximate solutions is equivalent to $v'(0; u) = \mathbf{q}(u)$. The following example shows that it is no longer the case in the absence of regularity. It also shows that $dv(0; u)$ may differ from $v'(0; u)$ in such cases.

EXAMPLE. Minimize x^2 subject to $u(x|x|^\alpha + u^\beta) \leq 0; \alpha > 1, \beta \geq 1, u \geq 0$. We have the following: $x = 0$ is the only solution of $\mathbf{P}(0), \Lambda(0) = \{(1, \lambda): \lambda \geq 0\}, \Lambda_u(0) = \Lambda(0)$,

$\mathbf{q}(u) = 0$, and both (**SQG**) are satisfied. For $u = 1$ the solution of $\mathbf{P}(tu)$ is $x(t) = -t^{\beta/(\alpha+1)}$ and $v(tu) = t^{2\beta/(\alpha+1)}$, so for $2\beta = \alpha + 1$ we have $v'(0;u) = 1 \neq \mathbf{q}(u) = dv(0;u)$; for $1 < \beta/(\alpha+1) < 2$ we have $v'(0;u) = \mathbf{q}(u)$ but no Lipschitz dependence of the solution on t at 0; for $\beta/(\alpha+1) < 1/2, u$ is not first-order feasible and even (4.17) is not valid. We observe finally that for any $\varepsilon > 0$, zero belongs to $S(tu, t\varepsilon)$ if t sufficiently small, so $dv(0;u) = 0 = \mathbf{q}(u)$, as was mentioned.

We also observe, in connection with the example, that if $\mathbf{q}(u) = \infty$, it may happen that the distance between $S(u)$ and S does not go to zero as $u \to 0$. It seems that little is known about such "singular" perturbations except for convex problems (see Gaitzgory and Pervozvanskii [25]).

5.5. Hölder stability of exact solutions as in Theorem 4.12 in presence of the standard second-order sufficient condition and (DR) was established by Gauvin and Janin [26], [27]. That constraint qualifications are rather irrelevant to properties of that kind is implicit in Bonnans [5], who proved that the conclusion of Theorem 4.12 is valid in a fully abnormal problem, provided the standard second-order sufficient condition is satisfied. (As this condition is symmetric with respect to the cost and constrained functions, the theorem of Bonnans is an easy corollary of Theorem 3.5.)

5.6. The upper estimates for the directional derivative of the value function given by $\mathbf{q}(u)$ and (4.11) have been known, the first for a long time (and present probably in all works on sensitivity analysis), and the second since [26]. The formula (4.9) for $\mathbf{a}(u)$ was found only recently, independently by Bonnans, Gfrerer, and Ioffe. The advantage of (4.9) is that it makes it possible to prove Theorem 4.9.

5.7. In a somewhat different context the idea to replace the second-order sufficient condition by a growth-like condition was used in [1], [2], [40], [62], and [68] in proofs of Lipschitz or Hölder stability of solutions of the perturbed problem. Namely, it was assumed that $f_o(y,0) \geq f_o(x,0) + \|y-x\|^\alpha$ for all admissible y (with $\alpha = 1$ or 2), and the purpose was to obtain a mechanism to analyze stability in the absence of smoothness or in the infinite-dimensional case. We observe that this condition was (and actually can only be) applied together with the regularity assumption, in which case for smooth finite-dimensional problems it is equivalent to the standard second-order sufficient condition.

The growth conditions (QG) and (LG) are substantially weaker (e.g., they do not imply by themselves that x is a local minimum; it is also clear that the above condition implies the corresponding growth condition), and the purpose of the growth condition is to obtain a mechanism to analyze stability in the case of arbitrary sets of solutions under no regularity assumptions, no matter whether the problem is smooth or not.

5.8. Sensitivity analysis of Lagrange multipliers remains beyond the scope of this article. We refer to [6], [28], [68], and [69] for some of the latest related results. It seems, however, that there is a certain asymmetry between multipliers and solutions that does not allow to prove, for multipliers, exact analogs of theorems we have for solutions.

On the other hand, Rockafellar [64] announced a theorem (with a reference to a forthcoming article by him and Poliquin) showing that under a suitable extension of the space of perturbation a very symmetric behavior is exhibited by *critical points*, on one hand, Lagrange multipliers, on the other, in the composite optimization problem involving composite functions of a certain class (called amenable) that is substantially broader than the class of composite functions considered here.

Acknowledgment. I am thankful to the referees for helpful remarks.

REFERENCES

[1] W. Alt, *Lipschitzian perturbations of infinite optimization problems,* in Mathematical Programming with Data Perturbations, A. V. Fiacco ed., Marcel Dekker, New York, 1983, pp. 7–21.

[2] A. Auslender, *Stability in mathematical programming with non-differentiable data,* SIAM J. Control Optim., 22 (1984), pp. 239–254.

[3] A. Auslender and R. Cominetti, *First and second order sensitivity conditions,* Optimization, 21 (1990), pp. 1–13.

[4] J. H. Bigelow and N. Z. Shapiro, *Implicit function theorems for mathematical programming and for system of inequalities,* Math. Programming, 6 (1974), pp. 141–156.

[5] J. F. Bonnans, *Directional derivatives of optimal solutions in smooth nonlinear programming,* J. Optim. Theory Appl., to appear.

[6] J. F. Bonnans, A. Ioffe, and A. Shapiro, *Développement de solutions exactes et approchées en programmation non linéaire,* C.R. Acad. Sci. Paris, 1991.

[7] ———, *On the general second order conditions for a minimum,* in preparation.

[8] J. Burke, *Second order necessary and sufficient condition for a composite NDO,* Math. Programming, 38 (1987), pp. 495–511.

[9] J. Burke and R. Poliquin, *Optimality conditions for non-finite composite functions,* Math. Programming, to appear.

[10] F. H. Clarke, *Optimization and Nonsmooth Analysis,* Wiley, New York, 1983.

[11] F. H. Clarke and P. Loewen, *The value function in optimal control: sensitivity, controllability and time optimality,* SIAM J. Control Optim., 24 (1986), pp. 243–263.

[12] R. Cominetti, *On pseudo differentiability,* Trans. Amer. Math. Soc., 324 (1991), pp. 843–866.

[13] F. Colonius and K. Kunish, *Sensitivity analysis for optimization problems in Hilbert spaces with bilateral constraints,* to appear.

[14] R. Correa and A. Seeger, *Directional derivatives of a minimax functions,* Nonlinear Anal. Theory, Methods, Appl., 9 (1985), pp. 13–22.

[15] J. M. Danskin, *The Theory of Max Min,* Springer-Verlag, Berlin, 1967.

[16] M. C. Delfour and J.-P. Zolésio, *Shape sensitivity analysis via Min-Max differentiability,* SIAM J. Control Optim., 26 (1988), pp. 834–862.

[17] V. F. Dem'janov, *Minimax: Directional Differentiability,* Leningrad St. Univ. Press, 1974. (In Russian.)

[18] V. F. Dem'janov and A. B. Pevnyi, *Marginal values of problems of mathematical programming and minimax problems,* Vestnik Leningrad Univ. Math., 6 (1979), pp. 339–347.

[19] S. Dolecki and S. Rolewicz, *Exact penalties for local minima,* SIAM J. Control Optim., 17 (1979), pp. 596–606.

[20] A. V. Fiacco, *Sensitivity analysis for nonlinear programming using penalty methods,* Math. Programming, 10 (1976), pp. 287–311.

[21] ———, *Introduction to Sensitivity and Stability Analysis in Nonlinear Programming,* Academic Press, New York, 1983.

[22] A. V. Fiacco and Yo. Ishizuka, *Sensitivity and stability analysis for nonlinear programming,* Ann. Oper. Res., 27 (1991), pp. 215–235.

[23] R. Fletcher, *Practical Methods of Optimization,* 1987.

[24] R. Fletcher and E. Saintz de Maza, *Nonlinear programming and nonsmooth optimization by successive linear programming,* Math. Programming, 43 (1989), pp. 257–276.

[25] V. A. Gaitzgory and A. A. Pervozvanskii, *Perturbation theory for mathematical programming problems,* J. Optim. Theory Appl., 49 (1986), pp. 389–410.

[26] J. Gauvin and R. Janin, *Directional behavior of optimal solutions in nonlinear mathematical programming,* Math. Oper. Res, 13 (1988), pp. 629–649.

[27] J. Gauvin, *Directional derivative for the value function in mathematical programming,* in Nonsmooth Optimization and Related Topics, F. Clarke, V. Den'janov, and F. Gianessi, eds., Plenum Press, New York, 1989, pp. 167–184.

[28] H. Gfrerer, *Directional behavior of stationary solution in nonlinear programming problems,* to appear.

[29] B. Gollan, *On the marginal value function in mathematical programming,* Math. Oper. Res., 9 (1984), pp. 208–221.

[30] J. Guddat, H. Jongen, B, Kummer, and F. Nožička, *Parametric Optimization and Related Topics,* Academie-Verlag, Berlin, 1987.

[31] J. Guddat, H. Jongen, and J. Rueckmann, *On stability and stationary points in nonlinear optimization,* J. Austral. Math. Soc. Ser B, 28 (1986), pp. 36–56.

[32] S. P. Hahn and O. Mangasarian, *Exact penalty functions in nonlinear programming,* Math. Programming, 17 (1979), pp. 251–269.

[33] A. D. Ioffe, *Necessary and sufficient conditions for a local minimum 1–3,* SIAM J. Control Optim., 17 (1979), pp. 245–288.

[34] A. D. Ioffe, *Second order conditions for nonlinear nonsmooth problems of semi-infinite programming,* in Semi-Infinite Programming and Applications, A. V. Fiacco and K. O. Kortanek, eds., Lecture Notes in Econ. and Math. Systems, Vol. 215, Springer-Verlag, Berlin, 1983.

[35] ———, *On some recent developments in the theory of second order optimality conditions,* in Optimization, S. Dolecki, ed., LNM, Vol 1405, Springer-Verlag, Berlin, 1989.

[36] ———, *Variational analysis of a composite function: a formula for the lower second order epi-derivative,* J. Math. Anal. Appl., to appear.

[37] A. D. Ioffe and V. M. Tihomirov, *Theory of Extremal Problems,* Nauka, Moscow, 1974. (In Russian.) English translation, 1979, North Holland, Amsterdam.

[38] K. Jittornum, *Solution point differentiability without strict complimentarity in nonlinear programming,* Math. Programming, 21 (1984), pp. 127–138.

[39] H. Th. Jongen, *Parametric optimization: Critical points and local minima,* Lectures in Appl. Math., 26 (1990), pp. 317–335.

[40] H. Th. Jongen, P. Jonker, and F. Twilt, *Critical sets in parametric optimization,* Math. Programming, 34 (1986), pp. 333–353.

[41] H. T. Jongen and G. W. Weber, *On parametric nonlinear programming,* Ann. Oper. Res., 27 (1991), pp. 253–283.

[42] D. Klatte, *On stability of local and global optimal solutions in parametric problems of nonlinear programming,* Seminarbericht no. 75, Humboldt-Universität, Berlin, 1985.

[43] M. Kojima, *Strongly stable stationary solutions in nonlinear programs,* in Analysis and Computation of Fixed Points, S. M. Robinson, ed., Academic Press, New York, 1980, pp. 93–138.

[44] M. Kojima and R. Hirabayashi, *Continuous deformations of nonlinear programs,* Math. Programming Stud., 21 (1984), pp. 150–198.

[45] F. Lempio and H. Maurer, *Differential stability in infinite dimensional nonlinear programming,* Appl. Math. Optim., 6 (1980), pp. 139–152.

[46] E. S. Levitin, *On the local perturbation theory of a problem of mathematical programming in a Banach space,* Soviet Math. Dokl., 16 (1975), pp. 1354–1358.

[47] E. S. Levitin, A. A. Miljutin, and N. K. Osmolovskii, *Higher order conditions for a local minimun in problems with constraints,* Russian Math. Surveys, 33 (1978), p. 6.

[48] K. Malanowski, *Sensitivity analysis of optimization problems in Hilbert space with application to optimal control,* Appl. Math. Optim., 21 (1990), pp. 1–20.

[49] J. P. Penot, *Second order conditions in composite optimization,* to appear.

[50] T. Pietrzykowski, *An exact potential method for constrained maxima,* SIAM J. Numer. Anal., 6 (1969), pp. 299–304.

[51] R. Poliquin, *An extension of Attouch's theorem and its application to second order epi-differentiation of convex composite functions,* to appear.

[52] R. Poliquin and R. T. Rockafellar, *Differentiability of solution mappings and sensitivity analysis in parametric optimization,* to appear.

[53] ———, *A calculus of epi-derivatives with application to optimization,* to appear.

[54] A. B. Poore, *Bifurcations in parametric pp. nonlinear optimization,* Ann. Oper. Res., 27 (1991), 343–369.

[55] S. M. Robinson, *Perturbed Kuhn–Tucker points and rates of convergence for a class of nonlinear programming algorithms,* Math. Programming, 7 (1974), pp. 1–16.

[56] ———, *Stability theory for systems of inequalities, Part 2: Differentiable nonlinear systems,* SIAM J. Numer. Anal., 13 (1976), pp. 497–513.

[57] ———, *Strongly regular generalized equations,* Math. Oper. Res., 5 (1980), pp. 43–62.

[58] ———, *Generalized equations and their solutions, Part* 2: *Applications to nonlinear programming,* Math. Programming Stud., 19 (1982), pp. 200–221.

[59] R. T. Rockafellar, *Convex Analysis,* Princeton Univ. Press, Princeton, NJ, 1970.

[60] ———, *Directional differentiability of the optimal value function in a nonlinear programming problem,* Math. Programming Stud., 21 (1984), pp. 213–226.

[61] ———, *First and second order epi-differentiability in nonlinear programming,* Trans. Amer. Math. Soc., 307 (1988), pp. 75–108.

[62] ———, *Second order optimality conditions in nonlinear programming obtained by way of epi-derivatives,* Math. Oper. Res., 14 (1989), pp. 462–484.

[63] ———, *Perturbations of generalized Kuhn–Tucker points in finite dimensional optimization,* in Nonsmooth Optimization and Related Topics, F. Clarke, V. Dem'janov, and F. Gianessi, eds., Plenum Press, New York, 1989.

[64] ———, *Nonsmooth analysis and parametric optimization,* to appear.

[65] W. Römisch and R. Schultz, *Distribution sensitivity in stochastic programming,* Math. Programming, 50 (1991), pp. 197–226.

[66] A. Shapiro, *Sensitivity analysis of nonlinear programs and differentiability properties of metric projections,* SIAM J. Control Optim., 26 (1988), pp. 628–645.

[67] A. SHAPIRO, *Perturbation theory of nonlinear programs when the set of optimal solutions is not a singleton,* Appl. Math. Optim., 18 (1988), pp. 215–229.
[68] ———, *Perturbation analysis of optimization problems in Banach spaces.*
[69] A. SHAPIRO AND J. F. BONNANS, *Sensitivity analysis of parameterized programs under cone constraints,* preprint, 1990.
[70] C. A. TIARHRT AND A. B. POORE, *A bifurcation analysis of the nonlinear parametric programming problem,* Math. Programming, 47 (1990), pp. 117–141.
[71] A. WIERZBICKI AND S. KURCYUSZ, *Projection on a cone, penalty functionals and duality theory for problems with inequality constraints in Hilbert spaces,* SIAM J. Control Optim., 15 (1977), pp. 25–56.

SIAM J. OPTIMIZATION
Vol. 4, No. 1, pp. 44–64, February 1994
© 1994 Society for Industrial and Applied Mathematics

A NONCONVEX DUALITY WITH ZERO GAP AND APPLICATIONS*

PHAN THIEN THACH†

Abstract. A duality with zero gap for nonconvex optimization problems is presented. The first class of nonconvex problems, where local optima may not be global, is a quasi-convex minimization over a convex set. For this class a generalized Kuhn–Tucker condition is obtained, and the duality is similar to the Fenchel–Moreau–Rockafellar duality scheme. By the duality, one can reduce the problem to solving a system of convex and quasi-convex inequalities. Unlike the previous developments, these conjugation functionals and dual problems are defined on the dual space and involve no extra parameter. For more general nonconvex problems, such as a quasi-convex maximization over a compact set or a general minimization over the complement of a convex set, a duality with zero gap can be obtained as well. A zero gap in primal-dual pairs allows the development of primal-dual algorithms for nonconvex problems. The primal-dual algorithms are very suitable when the dual problem is simpler than the primal one.

Key words. nonconvex duality with zero gap

AMS subject classifications. 49N15, 49J52, 90C26

1. Introduction. In this paper we present a duality with zero gap for nonconvex optimization problems. The presented nonconvex duality is constructed on quasi-conjugate functionals and quasi subdifferentials. The quasi-conjugate functionals, the quasi subdifferentials, and the dual problems are defined on the usual dual space. The first class of nonconvex problems is a quasi-convex minimization over a convex set:

$$\min\{f(x) : x \in D\}, \tag{1}$$

where f is a quasi-convex function and D a closed convex set. Since f is nonconvex, problem (1) may have local minima that are not global minima. It is well known that if f is a proper closed convex function, then a necessary and sufficient condition for the optimality of a feasible solution $z \in D$ is

$$0 \in \partial f(z) + N(z, D), \tag{2}$$

where $\partial f(z)$ denotes the set of subgradients of f at z and $N(z, D)$ denotes the normal cone of D at z. In convex programs a local optimal is global as well. When f is no longer convex, the condition (2), where $\partial f(z)$ is the set of the generalized subgradients of f at z, is inappropriate. In this paper we see that if $\partial f(z)$ is replaced by a convex set of the so-called quasi subdifferentials, a condition of the type (2) can be employed for problem (1) where local optima may not be global ones. There are close relationships between the so-called quasi subdifferentials and the subgradients in the class of convex functions. The basic mathematical material behind conditions of the type (2) is the separation theorem. This is natural because the optimality of a feasible solution $z \in D$ is equivalent to the nonintersection between two convex sets: D and $\{x : f(x) < f(z)\}$. Based on the quasi subdifferentials and the quasi-conjugate functionals, we obtain a dual problem with zero gap for a quasi-convex minimization over a closed convex set. The dual problem is defined in the usual dual space. The duality is similar to the Fenchel–Moreau–Rockafellar duality scheme. By the duality one can reduce problem (1) to solving a system of convex and quasi-convex inequalities. By this result a simple proof is obtained to prove that the complexity

* Received by the editors July 1, 1991; accepted for publication (in revised form) October 2, 1992.

† Institute of Human and Social Sciences, Tokyo Institute of Technology, 2-12-1 Oh-Okayama, Megura-ku, Tokyo 152, Japan. This paper was prepared while the author was visiting the Department of Mathematics, Trier University, Germany as a fellow of the Alexander von Humboldt Foundation.

of a problem of minimizing a quasi-convex function over a convex set is equivalent to the oracle problems (see Grotschel, Lovasz, and Schrijver [6] and Lovasz [11]). In more general problems, such as quasi-convex maximization over a compact set or a general minimization over the complement of a convex set, a duality with zero gap is also obtained. However, the optimality criterions are basically different from the type (2) and the duality scheme is basically different from the Fenchel–Moreau–Rockafellar duality scheme. This reflects the fact that one could not separate two nonconvex sets by a hyperplane.

The paper consists of nine sections. In §2 we draw definitions of quasi conjugates and quasi subdifferentials. In §3 we present a relationship between the quasi subdifferentials and the decreasing directions. In §4 we present a generalized Kuhn–Tucker condition for a quasi-convex minimization over a closed convex set. In §5 we present a duality scheme for a quasi-convex minimization over a closed convex set. In §6 we present a global optimality criterion and a dual problem for a quasi-convex maximization over a compact set. In §7 we present a global optimality criterion and a dual problem for a general minimization over the complement of a convex set. An application is given in §8. Finally we give some discussions and conclusions in §9.

2. Quasi conjugates and quasi subdifferentials of functions. Let $f : R^n \to R \cup \{\pm\infty\}$ be a function.

DEFINITION 2.1. A function f^H defined as follows

$$f^H(y) = \begin{cases} -\inf\{f(x) : \langle y, x\rangle \geq 1\} & \text{if } y \neq 0, \\ -\sup\{f(x) : x \in R^n\} & \text{if } y = 0, \end{cases}$$

is called the quasi conjugate of f. The biquasi conjugate of f, denoted by f^{HH}, is the quasi conjugate of the quasi conjugate of f.

The quasi conjugate was used in [24]. The function f^H is quasi-convex, and if f is quasi-convex, then $f^H(0) = \inf\{f(x) : x \neq 0\}$. Some of the basic properties of the quasi-conjugate functional are mentioned here.

Denote by Γ^U the set of upper semicontinuous (usc) functions $f : R^n \to R \cup \{\pm\infty\}$ such that $f(0) = \inf\{f(x) : x \in R^n\}$, by Γ^L the set of lower semicontinuous (lsc) functions $f : R^n \to R \cup \{\pm\infty\}$ such that $f(0) = \inf\{f(x) : x \in R^n \backslash \{0\}\}$, and

$$f(x) \to \sup\{f(x) : x \in R^n\} \quad \text{as } \|x\| \to \infty,$$

and by Γ the intersection $\Gamma^U \cap \Gamma^L$.

THEOREM 2.1. (i) *If $f \in \Gamma^U$ then $f^H \in \Gamma^L$;*

(ii) *If $f \in \Gamma^L$ then $f^H \in \Gamma^U$;*

(iii) *If $f \in \Gamma^L \cup \Gamma^U$ then f^{HH} is the greatest quasi-convex function majorized by f.*

Proof. (i) See Theorems 3.2, 3.3, and Corollary 4.2 in [24].

(ii) See Theorems 3.2 and 3.3 in [24].

(iii) See Theorems 4.1 and 4.2 in [24]. □

From Theorem 2.1 one can immediately obtain the following corollary.

COROLLARY 2.1. (i) *If $f \in \Gamma$ then $f^H \in \Gamma$;*

(ii) *If $f \in \Gamma$ then f^{HH} is the greatest quasi-convex function majorized by f;*

(iii) *If $f \in \Gamma$ and f is a quasi-convex, then $f^{HH} = f$.*

Now we give some relationships between the quasi conjugate and the conjugate of a function. We recall here the definition of the conjugate f^* of a function f:

$$f^*(y) = \sup_{x \in R^n} \{\langle y, x\rangle - f(x)\}.$$

THEOREM 2.2. (i) *For an arbitrary function* $f : R^n \to R \cup \{\pm\infty\}$, *one has*

$$f^H(y) \le f^*(y) - 1 \quad \forall y \in R^n \backslash \{0\}.$$

(ii) *If* f *is a finite convex function, then for every* $y \in R^n \backslash \{0\}$ *either*

$$f^H(y) = f^*(y) = \infty$$

or there is $t_0 \ge 0$ *such that*

$$f^H(y) = f^*(t_0 \cdot y) - t_0 = \inf_{t \ge 0}\{f^*(t \cdot y) - t\}. \tag{3}$$

Proof. (i) One has

$$\begin{aligned} f^H(y) &= -\inf\{f(x) : \langle y, x\rangle \ge 1\} \\ &= \sup\{-f(x) : \langle y, x\rangle \ge 1\} \\ &= \sup_x \inf_{t \ge 0}\{t(\langle y, x\rangle - 1) - f(x)\} \\ &= \sup_x \inf_{t \ge 0}\{\langle ty, x\rangle - t - f(x)\} \\ &\le \inf_{t \ge 0} \sup_x\{\langle ty, x\rangle - t - f(x)\} \\ &\le \sup_x\{\langle y, x\rangle - 1 - f(x)\} \\ &= \sup_x\{\langle y, x\rangle - f(x)\} - 1 \\ &= f^*(y) - 1. \end{aligned}$$

(ii) From assertion (i), if $f^H(y) = \infty$, then $f^*(y) = \infty$ as well. Now we suppose that $f^H(y) < \infty$, i.e.,

$$\inf\{f(x) : \langle y, x\rangle \ge 1\} > -\infty.$$

Then, there is $t_0 \ge 0$ such that

$$\begin{aligned} &\sup\{-f(x) : \langle y, x, \rangle \ge 1\} \\ &\quad = \sup_x\{\langle t_0 y, x\rangle - t_0 - f(x)\} \\ &\quad = \inf_{t \ge 0} \sup_x\{\langle ty, x\rangle - t - f(x)\} \end{aligned}$$

(see, e.g., Theorem 28.2 in Rockafellar [20]). Thus, we obtain (3). □

Let f be a lsc convex function. Denote by $\partial f(x)$ the set of subgradients of f at x. We know that $y \in \partial f(x)$ if and only if $f^*(y) + f(x) \le \langle y, x\rangle$. Thus, the subgradient of a convex function can be defined via the conjugate functional. In the sequel we give a definition of the quasi subdifferential via the quasi-conjugate functional.

Let $f : R^n \to R \cup \{\pm\infty\}$. For any $y \ne 0$ one has

$$f^H(y) = -\inf\{f(x) : \langle y, x\rangle \ge 1\} = \sup\{-f(x) : \langle y, x\rangle \ge 1\}.$$

So,

$$f^H(y) \ge -f(x) \quad \forall x : \langle y, x\rangle \ge 1. \tag{4}$$

DEFINITION 2.2. A vector y is called a quasi subdifferential of function f at x if

$$\langle y, x\rangle = 1 \quad \text{and} \quad f^H(y) \le -f(x).$$

The set of quasi subdifferentials of f at x is denoted by $\partial^H f(x)$. If $\partial^H f(x) \neq \emptyset$, we say that f is quasi-subdifferentiable at x. If $\partial^H f(x) \neq \emptyset$ for all $x \in X$, we say that f is quasi-subdifferentiable on X.

Because f^H is quasi-convex, if $\partial^H f(x)$ is nonempty, then $\partial^H f(x)$ is convex. From (4) $y \in \partial^H f(x)$ if and only if

$$\langle y, x\rangle = 1 \quad \text{and} \quad f^H(y) = -f(x).$$

We give now some properties of quasi subdifferentials and general conditions for the quasi subdifferentiability.

THEOREM 2.3. *For any $f : R^n \to R \cup \{\pm\infty\}$. One has*

$$\partial^H(\max(f(\cdot), \alpha))(x) = \partial^H f(x) \quad \forall\, \alpha < f(x). \tag{5}$$

Proof. Suppose that $y \in \partial^H f(x)$. Then, $\langle y, x\rangle = 1$ and

$$-f(x) = f^H(y) = -\inf\{f(z) : \langle y, z\rangle \ge 1\}.$$

For any $\alpha < f(x) = \inf\{f(z) : \langle y, z\rangle \ge 1\}$, one has

$$f(z) = \max\{f(z), \alpha\} \quad \forall\, z : \langle y, z\rangle \ge 1.$$

Therefore,

$$(\max\{f(\cdot), \alpha\})^H(y) = -\inf\{\max(f(\cdot), \alpha) : \langle y, x\rangle \ge 1\} = f^H(y).$$

This implies $y \in \partial^H(\max\{f(\cdot), \alpha\})$. Suppose that $y \in \partial^H(\max(f(\cdot), \alpha))(x)$ for some $\alpha < f(x)$. By definition, $\langle y, x\rangle = 1$ and

$$\begin{aligned} -f(x) &= \max(f(\cdot), \alpha)^H(y) \\ &= -\inf\{\max(f(z), \alpha) : \langle y, z\rangle \ge 1\}. \end{aligned}$$

Therefore,

$$\begin{aligned} &\max(f(z), \alpha) \ge f(x) \quad \forall\, z : \langle y, z\rangle \ge 1 \\ &\Rightarrow f(z) \ge f(x) \quad \forall\, z : \langle y, z\rangle \ge 1 \quad (\text{since } \alpha < f(x)) \\ &\Rightarrow -f(x) = -\inf\{f(z) : \langle y, z\rangle \ge 1\} = f^H(y). \end{aligned}$$

So, $y \in \partial^H f(x)$. □

THEOREM 2.4. (i) *If f is a finite convex function, then for every x such that $f(x) > f(0)$ one has $\partial^H f(x) \subseteq \cup_{t>0} t \cdot \partial f(x)$.*

(ii) *If f is a lsc convex function and $0 \in \text{int}(\text{dom } f)$ (dom $f = \{x : f(x) < \infty\}$), then for every $x \in$ dom f such that $f(x) > f(0)$ one has*

$$y \in \partial f(x) \Rightarrow (1/\langle y, x\rangle) \cdot y \in \partial^H f(x).$$

Proof. (i) Let $y \in \partial^H f(x)$. By the definition, $\langle y, x\rangle = 1$ and

$$-\infty < f(x) = -f^H(y) = \inf\{f(x) : \langle y, x\rangle \ge 1\}. \tag{6}$$

Therefore, by Theorem 2.2 there is $t \geq 0$ such that

$$f^H(y) = -\inf_x\{f(x) + t(1 - \langle y, x\rangle)\} = \sup_x\{\langle ty, x\rangle - f(x)\} - t \tag{7}$$
$$= f^*(ty) - t.$$

If $t = 0$, then $f^H(y) = f^*(0) = \sup\{-f(x) : x \in R^n\}$. This together with (6) implies $-f(x) \geq -f(0)$. We arrive at the contradiction with the fact that $f(0) < f(x)$. Thus, (7) happens with some $t > 0$. One has then

$$\begin{aligned}
&0 = f^H(y) + f(x) = f^*(ty) - t + f(x)\\
&\Leftrightarrow 0 = f^*(ty) - \langle ty, x\rangle + f(x) \quad (\text{since } \langle y, x\rangle = 1)\\
&\Leftrightarrow \langle ty, x\rangle = f^*(ty) + f(x)\\
&\Leftrightarrow ty \in \partial f(x)\\
&\Leftrightarrow y \in (1/t)\cdot \partial f(x).
\end{aligned}$$

So, $y \in \cup\{t \cdot \partial f(x) : t > 0\}$.

(ii) Let $y \in \partial f(x)$. Since $0 \in \text{int}(\text{dom } f)$ and $f(x) > f(0)$, one has $\langle y, x\rangle > 0$. So,

$$\begin{aligned}
f^H((1/\langle y, x\rangle)\cdot y) &= -\inf\{f(z) : \langle (1/\langle y, x\rangle)\cdot y, z\rangle \geq 1\}\\
&= -\inf\{f(z) : \langle y, x\rangle \geq \langle y, x\rangle\}.
\end{aligned}$$

Since $y \in \partial f(x)$, one has

$$f(z) \geq f(x) + \langle y, z - x\rangle \quad \forall z \in R^n.$$

Therefore,

$$f^H((1/\langle y, x\rangle)y) \leq -\inf\{f(x) + \langle y, z - x\rangle : \langle y, z\rangle \geq \langle y, x\rangle\} = -f(x).$$

By definition, $(1/\langle y, x\rangle)\cdot y \in \partial^H f(x)$. □

As a consequence of Theorem 2.4 one obtains the following corollary.

COROLLARY 2.2. *Let f be a finite convex function. For every x such that $f(x) > f(0)$ one has*

$$\text{cone } \partial^H f(x) = \text{cone } \partial f(x),$$

where cone$A = \{\lambda x : x \in A, \lambda \geq 0\}$.

THEOREM 2.5. *Let $f : R^n \to R \cup \{\pm\infty\}$ be an arbitrary function. If f is quasi-subdifferentiable at x, then $f^{HH}(x) = f(x)$.*

Proof. By definition, f is quasi-subdifferentiable at x if and only if $\partial^H f(x) \neq \emptyset$. So, there is $y \in \partial^H f(x)$:

$$\langle y, x\rangle = 1, f^H(y) = -f(x).$$

One has then

$$f(x) = -f^H(y) \leq -\inf\{f^H(v) : \langle v, x\rangle \geq 1\} = f^{HH}(x).$$

By Lemma 4.1 [24] this implies that $f(x) = f^{HH}(x)$. □

THEOREM 2.6. *Let $f \in \Gamma^U$. For any $x : f(x) > f(0)$ if $f^{HH}(x) = f(x)$, then f is quasi-subdifferentiable at x.*

Proof. Suppose that $\alpha = f^{HH}(x) = f(x) > f(0)$. Since f^{HH} is the greatest quasi-convex function majorized by f (see Theorem 2.1 (iii)), the set $G = \{z : f^{HH}(z) < \alpha\}$ is convex and contains the set $Q = \{z : f(z) < \alpha\}$. Since f^{HH} is usc (Theorem 2.1 (i) and (ii)), G is an open convex set. Since $x \notin G$, there is a linear function $\langle y, \cdot \rangle$ separating x and G:

$$\langle y, x \rangle > \langle y, z \rangle \quad \forall z \in G. \tag{8}$$

Since $0 \in Q \subseteq G$, this implies $\langle y, x \rangle > 0$. Therefore, without loss of generality, we can assume that $\langle y, x \rangle = 1$. From (8) and $Q \subseteq G$ this implies

$$f^*(y) = -\inf\{f(z) : \langle y, z \rangle \geq 1\} = -f(x). \tag{9}$$

Thus, $y \in \partial^H f(x)$ and hence f is quasi-subdifferentiable at x. □

COROLLARY 2.3. *If f is an usc quasi-convex function, then f is quasi-subdifferentiable on $\{x : f(x) > f(0)\}$.*

Proof. Set $g(x) = \max\{f(x), f(0)\}$. It is obvious that $g \in \Gamma^U$. By Theorem 2.1 (iii), one has $g^{HH} = g$. By Theorem 2.6, $\partial^H g(x) \neq \emptyset$ for all $x : g(x) > g(0)$. By Theorem 2.3, $\partial^H g(x) = \partial^H f(x)$ for all $x : f(x) > f(0)$. Thus, f is quasi-subdifferentiable on $\{x : f(x) > f(0)\}$. □

3. Quasi subdifferentials and decreasing directions. Unlike the usual subdifferential, the quasi subdifferential does not depend on the epigraph of a function. Indeed, we can easily check that $\partial^H(t \cdot f)(x) = \partial^H f(x)$ for all $t > 0$. In the sequel we shall see that the set of "decreasing" directions of f at x can be defined by $\partial^H f(x)$.

Let f be an usc function and x be a vector in R^n. Denote

$$F_x^< = \{z : f(z) < f(x)\},$$
$$H(f, x) = \{t \cdot (z - x) : z \in F_x^<, t > 0\}.$$

If $z \notin H(f, x)$, then the half-line $\{x + tz : t \geq 0\}$ does not intersect with $F_x^<$. So, $f(x) = \inf\{f(x + tz) : t \geq 0\}$. If $z \in H(f, x)$, then the half-line $\{x + tz : t \geq 0\}$ meets $F_x^<$. A vector $z \in H(f, x)$ is called a decreasing direction of f at x, and $H(f, x)$ is called the set of decreasing directions of f at x. Since $F_x^<$ is open, $H(f, x)$ is open. It is obvious that $H(f, x)$ is a cone.

THEOREM 3.1. *Let f be an usc function and x be a vector such that $f(x) > f(0)$. Function f is quasi-subdifferentiable at x if and only if*

$$0 \notin \text{conv}H(f, x). \tag{10}$$

If f is quasi-subdifferentiable at x, then $\partial^H f(x)$ is a compact convex set and

$$\text{conv}H(f, x) = \{z : \sup_{y \in \partial^H f(x)} \langle y, z \rangle < 0\}. \tag{11}$$

Proof. Suppose that (10) occurs. Since $\text{conv}H(f, x)$ is an open convex set, there is a linear function $\langle y, \cdot \rangle$ separating 0, and $\text{conv}H(f, x)$:

$$\langle y, 0 \rangle = 0 > \langle y, z \rangle \quad \forall z \in \text{conv}H(f, x).$$

Then, $0 > \langle y, z - x \rangle$ for all $z \in F_x^<$, or $\langle y, x \rangle > \langle y, z \rangle$ for all $z \in F_x^<$. Since $0 \in F_x^<$ and $F_x^<$ is open, this implies $\langle y, x \rangle > 0$. Without loss of the generality we can assume that $\langle y, x \rangle = 1$. Then, $1 = \langle y, x \rangle > \langle y, z \rangle$ for all $z \in F_x^<$. So, $-f^H(y) = \inf\{f(z) : \langle y, z \rangle \geq$

$1\} = f(x)$. This means $y \in \partial^H f(x)$. Now assume that f is quasi-subdifferentiable at x. Let $y \in \partial^H f(x)$. One has

$$\begin{aligned} & F_x^< \subseteq \{z : \langle y \cdot z\rangle < 1 = \langle y, x\rangle\} \\ \Rightarrow\ & F_x^< - x \subseteq \{z : \langle y, z\rangle < 0\} \\ \Rightarrow\ & H(f,x) \subseteq \{z : \langle y, z\rangle < 0\} \\ \Rightarrow\ & 0 \notin \operatorname{conv} H(f,x). \end{aligned}$$

In the sequel the second part of the theorem is proved. Set $g(x) = \max\{f(x), f(0)\}$. Since $f(x) > f(0)$, one has $G_x^< = F_x^<, H(f,x) = H(g,x)$, and $\partial^H f(x) = \partial^H g(x)$. Function g is usc and $g(0) = \inf\{g(z) : z \in R^n\}$. So, g^H is lsc and

$$g^H(y) \to \sup\{g^H(v) : v \in R^n\} \quad \text{as } \|y\| \to \infty \tag{12}$$

(see [24]). Since g^{HH} is the quasi-convex hull of g and $g(0) = \inf\{g(z) : z \in R^n\}$, one has

$$g(0) = g^{HH}(0) = -\sup\{g^H(v) : v \in R^n\}.$$

If $y \in \partial^H g(x)$, then

$$g^H(y) = -g(x) < -g(0) = \sup\{g^H(v) : v \in R^n\}.$$

This together with (12) implies that $\{v : g^H(v) \le -g(x)\}$ is compact. So, $\partial^H g(x) = \{x : \langle v \cdot x\rangle = 1, g^H(v) \le -g(x)\}$ is a compact convex set, and hence $\partial^H f(x)$ is a compact convex set. Let $z \in \operatorname{conv} H(f,x)$. Since $\partial^H f(x)$ is compact, there is $y \in \partial^H f(x)$ such that $\langle y, x\rangle = \sup\{\langle v, z\rangle : v \in \partial^H f(x)\}$. If

$$\langle y, z\rangle \ge 0,$$

then since $z \in \operatorname{conv} H(f,x)$ and $H(f,x)$ is open, one has

$$\begin{aligned} & \operatorname{conv} H(f,x) \cap \{u : \langle y, u\rangle \ge 0\} \ne 0 \\ \Rightarrow\ & H(f,x) \cap \{u : \langle y, u\rangle \ge 0\} \ne 0 \\ \Rightarrow\ & \{F_x^< - x\} \cap \{u : \langle y, u\rangle \ge 0\} \ne 0 \\ \Rightarrow\ & F_x^< \cap \{u : \langle y, u\rangle \ge \langle y, x\rangle = 1\} \ne 0 \\ \Rightarrow\ & -f^H(y) = \inf\{f(u) : \langle y, u\rangle \ge 1\} < f(x). \end{aligned}$$

We arrive at the contradiction with $y \in \partial^H f(x)$. So,

$$\langle y, x\rangle = \sup\{\langle v, z\rangle : v \in \partial^H f(x)\} < 1.$$

Let $z \notin \operatorname{conv} H(f,x)$. Since $0 \notin \operatorname{conv} H(f,x)$ and $\operatorname{conv} H(f,x)$ is a cone, one has $[0,z] \cap \operatorname{conv} H(f,x) = \emptyset$. Let $\langle y, \cdot\rangle$ be a linear function separating $[0,z]$ and $\operatorname{conv} H(f,x)$:

$$\inf_{u \in [0,z]} \langle y, u\rangle \ge \sup\{\langle y, v\rangle : v \in \operatorname{conv} H(f,x)\}. \tag{13}$$

Then,

$$\begin{aligned} & 0 \ge \sup\{\langle y, v\rangle : v \in \operatorname{conv} H(f,x)\} \\ \Rightarrow\ & 0 \ge \sup\{\langle y, v\rangle : v \in F_x^< - x\} \\ \Rightarrow\ & 0 \ge \sup\{\langle y, v - x\rangle : v \in F_x^<\} \\ \Rightarrow\ & \langle y, x\rangle \ge \sup\{\langle y, v\rangle : v \in F_x^<\}. \end{aligned}$$

Since $0 \in F_x^<$ and $F_x^<$ is open, one has $\langle y, x\rangle > 0$. Without loss of generality, we can assume that $\langle y, x\rangle = 1$. So, $1 = \langle y, x\rangle \geq \sup\{\langle y, v\rangle : v \in F_x^<\}$. This implies $F_x^< \subseteq \{v : \langle y, v\rangle < 1\}$. Therefore,

$$-f^H(y) = \inf\{f(v) : \langle y, v\rangle \geq 1\} = f(x).$$

Thus, $y \in \partial^H f(x)$. Further, from (13) one has

$$\langle y, x\rangle \geq \sup\{\langle y, v\rangle : v \in \operatorname{conv} H(f, x)\}.$$

Since $H(f, x)$ is a cone, this implies that $\langle y, z\rangle \geq 0$. So, $\sup\{\langle u, z\rangle : u \in \partial^H f(x)\} \geq \langle y, z\rangle \geq 1$. This completes the proof. □

Let f be an arbitrary usc function and x an arbitrary vector in R^n. If $\{z : f(z) < f(x)\} = \emptyset$, i.e., x is a minimizer of f on R^n, then $H(f, x) = \emptyset$. Otherwise, there is $w : f(w) < f(x)$. Set $g(x) = \max\{f(x + w), f(w)\}$. Then, $H(f, x) = H(g, x - w)$. Since g is usc and

$$g(x - w) = f(x) > f(w) = g(0),$$

the convex hull of $H(g, x - w)$ can be defined by $\partial^H g(x - w)$ (by Theorem 3.1). Therefore, the set of decreasing directions of a general usc quasi-convex function could be defined by the set of quasi-subdifferentials.

4. A global optimality criterion in a quasi-convex minimization over a convex set. Consider a problem

$$\min\{f(x) : x \in D\}, \tag{14}$$

where $f \in \Gamma$ is a quasi-convex function and D a closed convex set (we recall that Γ is a class of continuous functions $g : R^n \to R \cup \{\pm\infty\}$ satisfying $g(0) = \inf\{g(x) : x \in R^n\}$ and $g(x) \to \sup\{g(x) : x \in R^n\}$ as $\|x\| \to \infty$). Since $f \in \Gamma$ and D is closed, problem (14) has an optimal solution (see, e.g., Lemma 3.1 in [24]). We assume that

$$\sup\{f(x) : x \in R^n\} > \inf\{f(x) : x \in D\} > \inf\{f(x) : x \in R^n\}. \tag{15}$$

Since $f(0) = \inf\{f(x) : x \in R^n\}$, by Corollary 2.3, f is quasi-subdifferentiable on D. A vector $x \in D$ is called a global optimum if $f(x) = \inf\{f(x) : x \in D\}$ and a local optimum if there is a neighborhood M of x such that $f(x) \leq f(z)$ for all $z \in M \cap D$. Problem (14) may have many local optima that are not global optima.

Example 4.1.

$$f(t) = \begin{cases} |t| & \text{if } t \leq 1, \\ 1 & \text{if } 1 < t \leq 2, \\ t/2 & \text{if } 2 < t, \end{cases}$$

$$D = \{t : 5 \geq t \geq \tfrac{1}{2}\}.$$

Every $t \in [1, 2]$ is a local optimum but not a global optimum.

For a vector $x \in D$, we denote by $N(x, D)$ the normal cone of D at x:

$$N(x, D) = \{y : \langle y, z - x\rangle \leq 0 \ \forall z \in D\}.$$

THEOREM 4.1. *A sufficient condition for the global optimality of $x \in D$ is*

$$0 \in \partial^H f(x) + N(x, D). \tag{16}$$

Proof. Suppose that $x \in D$ and (16) occurs. There is a vector $y \in \partial^H f(x)$ such that $-y \in N(x, D)$. Since $y \in \partial^H f(x)$, one has $\langle y, x\rangle = 1$ and

$$f(x) = -f^H(y) = \inf\{f(z) : \langle y, z\rangle \geq 1\}. \tag{17}$$

Since $-y \in N(x, D)$, one has

$$\begin{aligned} &\langle y, z - x\rangle \geq 0 \quad \forall z \in D \\ \Leftrightarrow\ &\langle y, z\rangle \geq \langle y, x\rangle \quad \forall z \in D \\ \Leftrightarrow\ &\langle y, z\rangle \geq 1 \quad \forall z \in D. \end{aligned}$$

This together with (17) implies $f(x) = \inf\{f(z) : z \in D\}$. □

THEOREM 4.2. *Condition* (16) *is satisfied at at least a vector* $x \in D$.

Proof. Let $S = \{x : f(x) < \alpha\}, M = \{x \in D : f(x) = \alpha\}$. Then S is an open convex set and $0 \in S, M$ is a nonempty compact set. If $D \cap \text{cl}(S) \neq \emptyset$, then take $S' = S, x' \in D \cap \text{cl}(S)$. If $D \cap \text{cl}(S) = \emptyset$, then let $x' \in M$ such that

$$d(x', S) = \inf_{x \in M} d(x, S) = \inf\{\|x - y\| : x \in M, y \in S\}$$

and $S' = \{x' + t(y - x') : y \in S, 0 < t \leq 1\}$. In both cases, we can see that

$$0 \in S', \quad x' \in M, \quad S' \cap D = \emptyset, \quad \text{and} \quad x' \in \text{cl}(S').$$

Hence, there exist c and γ such that

$$\langle c, x\rangle < \gamma = \langle c, x'\rangle \leq \langle c, y\rangle \quad \forall x \in S', \quad \forall y \in D.$$

Since $0 \in S', \gamma$ is positive and can be taken equal to 1. Then, $-c \in N(x'D)$, because $\langle c, y - x'\rangle \geq 0$ for all $y \in D$, and $c \in \partial^H f(x')$, because $\langle c, x\rangle < 1$ whenever $f(x) < f(x')$. □

Example 4.2. We consider function f and set D defined as in Example 4.1. At every local minimum $t \in [1, 2]$, one has

$$\partial^H f(t) = \{1/t\}, \qquad N(t, D) = \{0\}.$$

So, condition (16) is not satisfied. At a unique global minimum $t = \frac{1}{2}$, condition (16) is satisfied:

$$\partial^H f(t) = \{2\}, \qquad N(t, D) = \{r : r \leq 0\}.$$

Assume that D is given by

$$D = \{x \in R^n : f_i(x) \leq 0,\ i = 1, \ldots, k\},$$

where $f_i : R^n \mapsto R$ are closed proper convex functions. If there exists $x \in R^n$:

$$f_i(x) < 0 \qquad \forall i = 1, \ldots, k,$$

then there exist nonnegative numbers $\lambda_1, \ldots, \lambda_k$ such that

$$N(x, D) = \text{cone}\left\{\sum_{i=1}^{k} \lambda_i \partial f_i(x)\right\},$$

and $\lambda_i = 0$ whenever $f_i(x) < 0$. In this case, condition (16) is equivalent to the fact that there exist nonnegative numbers $\lambda_1, \ldots, \lambda_k$ such that

$$0 \in \partial^H f(x) + \sum_{i=1}^{k} \lambda_i \partial f_i(x). \tag{18}$$

If f is a convex function, by Theorem 2.4 condition (18) is equivalent to

$$0 \in \partial f(x) + \sum_{i=1}^{k} \alpha_i \partial f_i(x) \tag{19}$$

for some $\alpha_i \geq 0, i = 1, \ldots, k$ and $\alpha_i = 0$ whenever $f_i(x) < 0$. This is the Kuhn–Tucker condition.

5. A duality scheme in quasi-convex minimization problems under a convex constraint. In the sequel we present a duality scheme for a quasi-convex minimization over a closed convex set. First, we consider the following definitions.

DEFINITION 5.1. Let $x \in D$ be a solution where condition (16) is satisfied. A vector $y \in -N(x, D) \cap \partial^H f(x)$ is called a generalized Kuhn–Tucker vector of problem (14) at x.

DEFINITION 5.2 (see, e.g., [29]). Let D be a closed convex set such that $0 \notin D$. The reverse polar of D denoted by D^H is the set defined as follows:

$$D^H = \{y \in R^n : \langle y, x \rangle \geq 1 \ \forall x \in D\}.$$

It is clear that the reverse polar D^H is a closed convex set and $0 \notin D^H$.

THEOREM 5.1 (see [29]). *For any closed convex set D such that $0 \notin D$, one has*

$$D^{HH} = \text{cl}\{t \cdot x : x \in D, t \geq 1\}. \tag{20}$$

THEOREM 5.2. *If $f \in \Gamma$ is a quasi-convex function, then for any closed convex set D one has*

$$\inf\{f(x) : x \in D\} = \inf\{f(x) : x \in \text{cl}\{tz : z \in D, t \geq 1\}\}. \tag{21}$$

Proof. Since $D \subseteq \text{cl}\{tz : z \in D, t \geq 1\}$, one has

$$\inf\{f(x) : x \in D\} \geq \inf\{f(x) : x \in \text{cl}\{tz : z \in D, t \geq 1\}\}. \tag{22}$$

Since $f \in \Gamma$, there is a minimizer v of f on $\text{cl}\{tz : z \in D, t \geq 1\}$ (see [24]). There are sequences $\{z^k\} \subset D$ and $\{t^k\}, t^k \geq 1$ such that $t^k \cdot z^k \to v$. Since $f(0) < f(z^k)$, from the quasi convexity of f one has $f(t^k \cdot z^k) \geq f(z^k)$. Therefore,

$$\inf\{f(x) : x \in D\} \leq \liminf f(z^k) \leq \liminf f(t^k \cdot z^k) = f(v).$$

This completes the proof. □

Now we call the dual problem of (14) the following problem:

$$\inf\{f^H(y) : y \in D^H\}. \tag{23}$$

THEOREM 5.3. *One has* $\inf(14) = -\inf(23)$.

Proof. For all $x \in D, y \in D^H$, one has $\langle y, x \rangle \geq 1$. So $-f^H(y) = \inf\{f(z) : \langle y, z \rangle \geq 1\} \leq f(x)$. Therefore, $\inf(14) \geq -\inf(23)$. It remains to prove the inverse inequality.

Let x be a solution of (14) and y be a generalized Kuhn–Tucker vector of (14) at x. Since $\langle y, x\rangle = 1$ and $-y \in N(x, D)$, y belongs to D^H. One then has $\inf(14) = f(x) = -f^H(y) \leq -\inf(23)$. □

We see that in the dual (23), f^H is quasi-convex and belongs to Γ, and further,

$$\sup\{f^H(y) : y \in R^n\} > \inf(23) > \inf\{f^H(y) : y \in R^n\}.$$

Indeed, this can be obtained from (15), Theorem 5.3, and the fact that

$$\sup\{f^H(y) : y \in R^n\} = -f^{HH}(0) = -f(0),$$
$$\inf\{f^H(y) : y \in R^n\} = f^H(0) = -\sup\{f(x) : x \in R^n\}.$$

The dual problem (23) belongs to the same class of the primal problem (14). If we take the dual of problem (23), then we obtain

$$\inf\{f^{HH}(x) : x \in D^{HH}\}.$$

Since $f^{HH} = f$, by Theorem 5.2 this problem is equivalent to the primal (14). Thus, the duality relationship is symmetric within the class of considered quasi-convex minimization problems.

COROLLARY 5.1. *A vector $(x, y) \in D \times D^H$ is a solution of the inequality*

$$f(x) + f^H(y) \leq 0$$

if and only if x is an optimal solution to (14) *and y is an optimal solution to* (23).

Proof. This corollary is immediately obtained from Theorem 5.3. □

Corollary 5.1 reduces problem (14) to find a vector in a convex set

$$A := \{(x, y) : x \in D, y \in D^H, f(x) + f^H(y) \leq 0\}.$$

If we assume that $f(x)$ can be polynomially computed and we have a polynomial subroutine to test if a given vector x belongs to D, then problem (14) is equivalent to the oracle problems (see [6] and [11]), i.e., if the oracles can be realized by polynomial algorithms, then problem (14) can. Indeed, to check if $(x, y) \in A$, we have to check if $x \in D, y \in D^H$, and $f(x) + f^H(y) \leq 0$. The problem of checking if $x \in D$, by assumption, can be realized by a polynomial subroutine. Since $D^H = \{y : \inf_{x \in D}\langle y, x\rangle \geq 1\}$, the problem of checking if $y \in D^H$ is the validity oracle. Moreover,

$$-f^H(y) < f(x) \Leftrightarrow \sup\{\langle y, z\rangle : f(z) < f(x)\} > 1.$$

Indeed, suppose that $-f^H(y) < f(x)$. By definition,

$$f(x) > \inf\{f(z) : \langle y, z\rangle \geq 1\}.$$

Since $f \in \Gamma$, there is z^* such that $\langle y, z^*\rangle \geq 1$ and

$$f(z^*) = \min\{f(z) : \langle y, z\rangle \leq 1\}$$

(see [24]). So,

$$\sup\{\langle y, z\rangle : f(z) < f(x)\} > \langle y, z^*\rangle = 1.$$

Conversely, suppose that $\sup\{\langle y, z\rangle : f(z) < f(x)\} > 1$. Then there is z^* such that $f(z^*) < f(x)$ and $\langle y, z^*\rangle > 1$. So,

$$f(x) > \inf\{f(z) : \langle y, z\rangle \geq 1\}$$

and hence $-f^H(y) < f(x)$. Thus,

$$-f^H(y) \geq f(x) \Leftrightarrow \sup\{\langle y, z\rangle : f(z) < f(x)\} \leq 1$$
$$\Leftrightarrow \langle y, z\rangle \leq 1 \qquad \forall z : f(z) < f(x).$$

Therefore, the problem of checking if $f^H(y) \leq -f(x)$ is the validity oracle. If the oracles can be solved in polynomial time, then the problem of checking if $(x, y) \in A$ can be solved in polynomial time. Because A is convex, the problem of finding a vector in A is an oracle.

Let z be a solution of (14) and y be a generalized Kuhn–Tucker vector of (14) at z. Then,

$$1 = \langle y, z\rangle \leq \langle y, x\rangle \quad \forall x \in D.$$

So, y belongs to D^H.

THEOREM 5.4. *Let $(z, y) \in D \times D^H$. The following assertions are equivalent*:

(i) *y is a generalized Kuhn–Tucker vector of the primal problem* (14) *at z*;

(ii) *z is a generalized Kuhn–Tucker vector of the dual problem* (23) *at y.*

Proof. The first assertion is equivalent to

$$1 = \langle y, z\rangle \leq \langle y, x\rangle \quad \forall x \in D, \tag{24}$$

$$f^H(y) = -f(z). \tag{25}$$

Since f is quasi-convex and $f \in \Gamma$, $f^{HH} = f$ (see Corollary 2.1). Therefore, (24) and (25) are equivalent to

$$1 = \langle y, z\rangle \leq \langle v, z\rangle \quad \forall v \in D^H \quad \text{and} \quad f^{HH}(z) = -f^H(y).$$

By definition, this is equivalent to the second assertion. □

COROLLARY 5.2. *A vector $(z, y) \in D \times D^H$ is a solution of the following system of inequalities*

$$\langle z, y\rangle \leq 1, \tag{26}$$

$$f(z) + f^H(y) \leq 0 \tag{27}$$

if and only if z is an optimal solution of the primal (14) *and y is a generalized Kuhn–Tucker vector of* (14) *at z.*

Proof. For all $(z, y) \in D \times D^H$ one has $\langle z, y\rangle \geq 1$, and $f(z) \geq -f^H(y)$. Thus, (26) and (27) are equivalent to $\langle z, y\rangle = 1$ and $f(z) = -f^H(y)$. This is equivalent to the fact that z is optimal to (14) and y is a generalized Kuhn–Tucker vector at z. □

6. Global optimality criterion and a dual problem for a quasi-convex maximization over a compact set. Consider a problem

$$\max\{f(x) : x \in D\}, \tag{28}$$

where $f \in \Gamma^U$ is quasi-convex and D is a compact set containing 0. Since f is usc, and D is compact, problem (28) has an optimal solution. We suppose that f is nonconstant on D. So,

$$f(0) = \inf\{f(x) : x \in R^n\} = \inf\{f(x) : x \in D\} < \max(28).$$

THEOREM 6.1. *A vector $z \in D$ is a global optimal solution to problem* (28) *if and only if f is quasi-subdifferentiable at z and for every $y \in \partial^H f(z)$ one has*

$$f^H(y) = \min\{f^H(v) : v \notin \text{int}D^0\}.$$

Since $f^H(y) = -f(z)$ for all $y \in \partial^H f(z)$, the term "for every $y \in \partial^H f(z)$" in this theorem is equivalent to the term "for a vector $y \in \partial^H f(z)$."

Proof. Suppose that $z \in D$ is optimal to problem (28). One has $f(z) = \max(28) > f(0)$. By Corollary 2.3, f is quasi-subdifferentiable at z. Let $y \in \partial^H f(z)$. Since $\langle y, z\rangle = 1, y$ does not belong to $\text{int}D^0$. Let $v \notin \text{int}D^0$. One has $\sup_{x\in D}\langle v, x\rangle \geq 1$. Therefore, there exists $x' \in D : \langle v, x'\rangle \geq 1$. Then,

$$f^H(v) = -\inf\{f(x) : \langle v, x\rangle \geq 1\} \geq -f(x') \geq -\max(28) = -f(z) = f^H(y).$$

So,

$$f^H(y) = \min\{f^H(v) : v \notin \text{int}D^0\}.$$

Suppose now that f is quasi-subdifferentiable at z and there exists $y \in \partial^H f(z)$ such that $f^H(y) = \min\{f^H(v) : v \notin \text{int}D^0\}$. One then has

$$\begin{aligned} f(z) &= -f^H(y) = -\inf\{f^H(v) : v \notin \text{int}D^0\} \\ &= -\inf_v\{-\inf_x\{f(x) : \langle v, x\rangle \geq 1\} : v \notin \text{int}D^0\} \\ &= \sup_v \inf_x\{\{f(x) : \langle v, x\rangle \geq 1\} : v \notin \text{int}D^0\} \\ &\geq \inf_x\{f(x) : \langle v, x\rangle \geq 1\} \quad \forall v \notin \text{int}D^0. \end{aligned} \tag{29}$$

Let z' be an optimal solution to problem (28), and $y' \in \partial^H f(z')$. Then, $y' \notin \text{int}D^0$ and

$$f(z') = -f^H(y') = \inf\{f(x) : \langle y', x\rangle \geq 1\}. \tag{30}$$

From (29) and (30) it follows that $f(z) \geq f(z')$. This completes the proof. □

We call the dual problem of problem (28) the following problem

$$\min\{f^H(v) : v \notin \text{int}D^0\}. \tag{31}$$

Since $f \in \Gamma^U$ and D is compact, $f^H \in \Gamma^L$ and $0 \in \text{int}D^0$. Furthermore, by Theorem 6.1, one has $\max(28) = -\min(31)$. Indeed, let z be an optimal solution to problem (28) and $y \in \partial^H f(z)$. One has

$$\max(28) = f(z) = -f^H(y) = -\min(31).$$

Thus we obtain the following corollary.

COROLLARY 6.1. (i) $\max(28) = -\min(31)$;

(ii) *A vector $z \in D$ is optimal to problem* (28) *if and only if $y \in \partial^H f(z)$ is optimal to the dual problem* (31).

7. Global optimality criterion and a dual problem for a general minimization over the complement of a convex set. Consider a problem

$$\min\{f(x) : x \notin \text{int}D\}, \tag{32}$$

where $f \in \Gamma^L$ and D is a closed convex set containing 0 in its interior. Since $f \in \Gamma^L$, problem (32) has an optimal solution (see Lemma 3.1 in [24]).

THEOREM 7.1. *A vector z is optimal to problem* (32) *if and only if $\partial^H f(z) \cap D^0 \neq \emptyset$ and for every $y \in \partial^H f(z)$ one has*

$$f^H(y) = \sup\{f^H(v) : v \in D^0\}.$$

Since $f^H(y) = -f(z)$ for all $y \in \partial^H f(z)$, the term "for every $y \in \partial^H f(z)$" in this theorem is equivalent to the term "for a vector $y \in \partial^H f(z)$."

Proof. One has

$$\begin{aligned}
&\sup\{f^H(v) : v \in D^0\} \\
&= \sup\{f^H(v) : v \in D^0 \backslash \{0\}\} \\
&= \sup\{-\inf\{f(x) : \langle v, x\rangle \geq 1\} : v \in D^0\} \\
&= -\inf_{v \in D^0} \inf_x \{f(x) : \langle v, x\rangle \geq 1\} \\
&= -\inf_x \inf_{v \in D^0} \{f(x) : \langle v, x\rangle \geq 1\}.
\end{aligned}$$

Since

$$\langle v, x\rangle \geq 1 \quad \text{for some } v \in D^0 \Leftrightarrow x \notin \text{int}D,$$

this implies

$$\sup\{f^H(v) : v \in D^0\} = -\inf\{f(x) : x \notin \text{int}D\}. \tag{33}$$

Let $z \notin \text{int}D$ be an optimal solution to problem (32). Since $z \notin \text{int}D$, there is a vector $y \in D^0$ such that $\langle y, z\rangle = 1$. Then,

$$\begin{aligned}
-f^H(y) &= \inf\{f(x) : \langle y, x\rangle \geq 1\} \\
&\geq \inf\{f(x) : x \notin \text{int}D\} \quad (\text{since } y \in D^0) \\
&= f(z).
\end{aligned}$$

So, by definition, $y \in \partial^H f(z)$. Therefore, $\partial^H f(z) \cap D^0 \neq 0$. Let $y \in \partial^H f(z)$. One has

$$f^H(y) = -f(z) = -\inf\{f(x) : x \notin \text{int}D\} = \sup\{f^H(v) : v \in D^0\}.$$

Now suppose that $\partial^H f(z) \cap D^0 \neq \emptyset$ and $f^H(y) = \sup\{f^H(v) : v \in D^0\}$ for some $y \in \partial^H f(z) \cap D^0$. Since $y \in D^0$ and $\langle y, z\rangle = 1$, one has $z \notin \text{int}D$. So, z is feasible to (32). Furthermore,

$$f(z) = -f^H(y) = -\sup\{f^H(v) : v \in D^0\} = \inf\{f(x) : x \notin \text{int}D\}.$$

So, z is optimal to (32). □

We call the dual problem of problem (32) the following problem

$$\max\{f^H(v) : v \in D^0\}. \tag{34}$$

Since $0 \in \text{int}D$ and $f \in \Gamma^L$, one has D^0 is compact and $f^H \in \Gamma^U$. Problem (34) is a quasi-convex maximization over a convex set.

COROLLARY 7.1. (i) $\min\{f(x) : x \notin \text{int}D\} = -\max\{f^H(v) : v \in D^0\}$;

(ii) *A vector z is optimal to problem* (32) *if and only if $\partial^H f(z) \cap D^0 \neq \emptyset$ and every vector $y \in \partial^H f(z) \cap D^0$ is optimal to the dual problem* (34).

8. An application. In R^n we are given $m+1$ closed convex sets: $X_0, \ldots, X_m$, such that $0 \in \text{int}X_0, 0 \notin X_i, i = 1, \ldots, m$. Consider the following problem.

Problem A. To find the biggest open ball centered at 0 that is contained in X_0 and does not intersect with $X_i, i = 1, \ldots, m$.

This problem can be formulated as follows

$$\min\{\inf\{\|x\| : x \notin \text{int}X_0\}, \inf\{\|x\| : x \in X_i\}, i = 1, \ldots, m\}. \tag{35}$$

To solve Problem A we have to solve m convex programs:

$$\inf\{\|x\| : x \in X_i\}, \tag{36}$$

$i = 1, \ldots, m$, and a convex minimization over the complement of a convex set:

$$\inf\{\|x\| : x \notin \text{int}X_0\}. \tag{37}$$

The quasi conjugate of function $f(x) = \|x\|$ is $f^H(y) = -1/\|y\|$. So, the dual of problem (36) is

$$\inf\{-1/\|y\| : y \in X_i^H\} \tag{38}$$

and the dual of problem (37) is

$$\sup\{-1/\|y\| : y \in X_0^0\}. \tag{39}$$

By the duality theorems one has $\inf(36) = -\inf(38)$, and $\inf(37) = -\sup(39)$. Therefore,

$$\begin{aligned}
&\min\{\inf\{\|x\| : x \notin \text{int}X_0\} \inf\{\|x\| : x \in X_i\}, i = 1, \ldots, m\} \\
&= \min\{-\sup\{-1/\|y\| : y \in X_0^0\}, -\inf\{-1/\|y\| : y \in X_i^H\}, i = 1, \ldots, m\} \\
&= -\max\{\sup\{-1/\|y\| : y \in X_0^0\}, \inf\{-1/\|y\| : y \in X_i^H\}, i = 1, \ldots, m\} \\
&= 1/\max\{\sup\{\|y\| : y \in X_0^0\}, \inf\{\|y\| : y \in X_i^H\}, i = 1, \ldots, m\}.
\end{aligned}$$

We call the dual problem of problem (35) the following problem:

$$\max\{\sup\{\|y\| : y \in X_0^0\}, \inf\{\|y\| : y \in X_i^H\}, i = 1, \ldots, m\}. \tag{40}$$

This problem is to find the smallest closed ball centered at 0 that contains X_0^0 and intersects with $X_i^H (i = 1, \ldots, m)$. By setting $Y_0 = X_0^0, Y_i = X_i^H$, we obtain the following dual problem of Problem A.

Problem B. To find the smallest closed ball centered at 0 that contains Y_0 and intersects with $Y_i, i = 1, \ldots, m$.

Similarly, by taking the dual problems of problems (38) and (39), we obtain the dual problem of Problem B:

$$\min\{\inf\{\|x\| : x \notin \text{int}Y_0^0\}, \inf\{\|x\| : x \in Y_i^H\}, i = 1, \ldots, m\}. \tag{41}$$

Since $0 \in X_0$, one has $Y_0^0 = X_0^{00} = X_0$. Since $0 \notin X_i$, by Theorem 5.1 one has $Y_i^H = X_i^{HH} = \text{cl}\{t \cdot x : x \in X_i, t \geq 1\}$. By Theorem 5.2,

$$\inf\{\|x\| : x \in X_i\} = \inf\{\|x\| : x \in Y_i^H\}.$$

So, an open ball does not intersect with X_i if and only if it does not intersect with $Y_i^H = X_i^{HH}$. Therefore, problem (41) is exactly Problem A.

In the sequel two parametric problems associated with Problems A and B are considered.

Set

$$\begin{aligned} h_0(x) &= \sup\{\langle y, x\rangle : y \in Y_0\}, \\ g_0(y) &= \sup\{\langle y, x\rangle : x \in X_0\}, \\ h_i(x) &= \inf\{\langle x, y\rangle : y \in Y_i\}, \qquad i = 1, \ldots, m, \\ g_i(y) &= \inf\{\langle y, x\rangle : x \in X_i\}, \qquad i = 1, \ldots, m. \end{aligned}$$

Functions h_0, g_0 are homogenous convex and functions h_i, g_i are homogenous concave. One has

$$\begin{aligned} X_0^0 &= \{y : g_0(y) \le 1\}, \\ Y_0^0 &= \{x : h_0(x) \le 1\}, \\ X_i^H &= \{y : g_i(y) \ge 1\}, \qquad i = 1, \ldots, m, \\ Y_i^H &= \{x : h_i(x) \ge 1\}, \qquad i = 1, \ldots, m. \end{aligned}$$

Problem (41) (or, equivalently, Problem A) can be rewritten as

$$\min\{\inf\{\|x\| : h_0(x) \ge 1\}, \inf\{\|x\| : h_i(x) \ge 1\}, i = 1, \ldots, m\}, \tag{42}$$

and problem (40) (or, equivalently, Problem B) can be rewritten as

$$\max\{\sup\{\|y\| : g_0(y) \le 1\}, \inf\{\|y\| : g_i(y) \ge 1\}, i = 1, \ldots, m\}. \tag{43}$$

Consider the following parametric problems:

$$\begin{aligned} (A_u) \qquad & \min\{\inf\{\|x\| : h_0(x) \ge 1 + \langle u, x\rangle\}, \\ & \qquad \inf\{\|x\| : h_i(x) \ge 1 + \langle u, x\rangle\}, i = 1, \ldots, m\}, \\ (B_v) \qquad & \max\{\sup\{\|y\| : g_0(y) \le 1 + \langle v, y\rangle\}, \\ & \qquad \inf\{\|y\| : g_i(y) \ge 1 + \langle v, y\rangle\}, i = 1, \ldots, m\}, \end{aligned}$$

where u, v are vectors in R^n. It is obvious that problems (A_u) with $u = 0$ and (B_v) with $v = 0$ are Problems A and B, respectively. For each u and v denote by $r(u)$ and $s(v)$ the optimal values in (A_u) and (B_v), respectively. We are now interested in maximizing $r(u)$ and minimizing $s(v)$:

$$\sup\{r(u) : u \in R^n\}, \tag{44}$$

$$\inf\{s(v) : v \in R^n\}. \tag{45}$$

Set

$$\varphi(u) := \max\{\sup_y\{\|u - y\| : y \in Y_0\}, \inf_y\{\|u - y\| : y \in Y_i\}, i = 1, \ldots, m\},$$

$$\phi(v) := \min\{\inf_x\{\|v - x\| : x \notin \operatorname{int} X_0\}, \inf_x\{\|v - x\| : x \in X_i\}, i = 1, \ldots, m\}.$$

The theorem below will establish the duality between the problems (44), (45), and the following design centering problems

$$\varphi(u) \to \inf, \quad \text{s.t. } u \in R^n \tag{46}$$

$$\phi(v) \to \sup, \quad \text{s.t. } v \in R^n. \tag{47}$$

Problem (46) is to find the smallest closed ball that contains Y_0 and intersects with Y_i, $i = 1, \ldots, m$, and problem (47) is to find the biggest open ball that is contained in X_0 and does not intersect with X_i. The applications of problems (46) and (47) can be found in [18].

THEOREM 8.1. (i) $r(u) = 1/\varphi(u)$;

(ii) $s(v) = 1/\phi(v)$.

Proof. Since

$$\begin{aligned}(Y_0 - u)^0 &= \{x : \langle x, y - u\rangle \le 1 \ \forall y \in Y_0\}\\ &= \{x : \langle y, x\rangle \le 1 + \langle x, u\rangle \ \forall y \in Y_0\}\\ &= \{x : h_0(x) \le 1 + \langle x, u\rangle\},\end{aligned}$$

one has

$$\{x : h_0(x) \le 1 + \langle x, u\rangle\}^0 = \{Y_0 - u\}^{00} = \operatorname{conv}(0, Y_0 - u).$$

By Corollary 7.1, this implies

$$\begin{aligned}&\inf\{\|x\| : h_0(x) \ge 1 + \langle x, u\rangle\}\\ &\quad = -\sup\{-1/\|y\| : y \in \operatorname{conv}(0, Y_0 - u)\}\\ &\quad = 1/\sup\{\|y\| : y \in \operatorname{conv}(0, Y_0 - u)\}\\ &\quad = 1/\sup\{\|y\| : y \in Y_0 - u\}\\ &\quad = 1/\sup\{\|y - u\| : y \in Y_0\}.\end{aligned} \tag{48}$$

If $u \in Y_i$ for some $i \in \{1, \ldots, m\}$, then

$$\begin{aligned}&\{x : h_i(x) \ge 1 + \langle x, u\rangle\}\\ &\quad = \{x : \inf_{y \in Y_i} \langle y, x\rangle \ge 1 + \langle u, x\rangle\}\\ &\quad = \{x : \inf_{y \in Y_i} \langle y, x\rangle - \langle u, x\rangle \ge 1\} = \emptyset\end{aligned}$$

and hence

$$\inf\{\|x\| : h_i(x) \ge 1 + \langle x, u\rangle\} = \infty = 1/0 = 1/\inf\{\|u - y\| : y \in Y_i\}. \tag{49}$$

If $u \notin Y_i$ for some $i \in \{1, \ldots, m\}$, then $0 \notin Y_i - u$, and hence

$$\begin{aligned}\{Y_i - u\}^H &= \{x : \langle y - u, x\rangle \ge 1 \ \forall y \in Y_i\}\\ &= \{x : \langle y, x\rangle \ge 1 + \langle u, x\rangle \ \forall y \in Y_i\}\\ &= \{x : h_i(x) \ge 1 + \langle u, x\rangle\}.\end{aligned}$$

So, by Theorem 5.3 one has

$$\begin{aligned}&\inf\{\|x\| : h_i(x) \ge 1 + \langle u, x\rangle\}\\ &\quad = -\inf\{-1/\|y\| : y \in Y_i - u\}\\ &\quad = 1/\inf\{\|y\| : y \in Y_i - u\}\\ &\quad = 1/\inf\{\|y - u\| : y \in Y_i\}.\end{aligned} \tag{50}$$

From (48)–(50) we obtain

$$\begin{aligned}r(u) &= \min\{\inf\{\|x\| : h_0(x) \ge 1 + \langle u, x\rangle\},\\ &\qquad \inf\{\|x\| : h_i(x) \ge 1 + \langle u, x\rangle\}, i = 1, \ldots, m\}\\ &= \min\{1/\sup\{\|y - u\| : y \in Y_0\},\ 1/\inf\{\|y - u\| : y \in Y_i\}, i = 1, \ldots, m\}\\ &= 1/\max\{\sup\{\|y - u\| : y \in Y_0\},\ \inf\{\|y - u\| : y \in Y_i\}, i = 1, \ldots, m\}\\ &= 1/\varphi(u).\end{aligned}$$

By analogous arguments we can prove that $s(v) = 1/\phi(v)$. □

By Theorem 8.1, instead of maximizing function $r(u)$ (respectively, minimizing function $s(v)$), we can minimize function $\varphi(u)$ (respectively, maximize function $\phi(v)$). The functions $\varphi(u)$ and $\phi(v)$ have Lipschitz constant 1 [23], whereas the functions $r(u)$ and $s(v)$ are not Lipschitzian. Further applications of the nonconvex duality with zero gap can be found in [27] and [28].

9. Conclusions. An important question in a nonconvex duality is how to generalize the conjugation and the subdifferentials for quasi-convex functions. Because the class of quasi-convex functions is much bigger than the class of convex functions, the quasi conjugations are usually defined on a space larger than the dual vector space (see, e.g., Greenberg and Pierskalla [5], Crouzeix [3], Atteia and El Qortobi [1], Oettli [14], [15], Singer [21], [22], Passy and Prisman [16], Martinez-Legaz [12], and Penot and Volle [17]). For a unified approach of generalizations readers are referred to the paper by Penot and Volle [17]. Since the epigraph of a quasi-convex function is the intersection of the epigraphs of the functionals that are the composition of nondecreasing extended real-valued functionals and linear functions, a subclass of nondecreasing functionals is often involved in the generalizations. For instance, let us recall the K-subdifferential defined in Penot and Volle [17]. For a subclass K of nondecreasing extended real-valued functions, the K-subdifferential $\partial^K f(x_0)$ of $f : R^n \mapsto R \cup \{\pm\infty\}$ at $x_0 \in R^n$ is the set of $y \in R^n$ such that there exists $k \in K$ with $k(\langle y, x_0\rangle) \in R$ and

$$f(x) \geq k(\langle y, x\rangle) - k(\langle y, x_0\rangle) + f(x_0) \qquad \forall\, x \in R^n.$$

If $\partial^K f(x_0) = \emptyset$, then f is not K-subdifferentiable at x_0. This definition of course depends on the class K. We often try to use a class K as small as possible. But there is, in fact, a contradiction between the simplicity of K and the K-subdifferentiability. If K is too small (for example, K is contained in the class of increasing functions), then quasi-convex functions are not K-subdifferentiable at the local minima that are not global. Unlike the previous generalizations, our results are presented on the more specific classes of quasi-convex functions (Γ, Γ^U, and Γ^L). We establish the optimality criterions and the duals for three types of nonconvex optimization problems: *q*uasi-convex *mi*nimization over a *c*onvex set (QMIC), *q*uasi-convex *ma*ximization over a compact *c*onvex set (QMAC), and *q*uasi-convex *mi*nimization over the *c*omplement of a *c*onvex set (QMICC). In our results the main restriction (comparing with the previous generalizations) is the assumption that the quasi-convex objective functions have a minimum value at the origin. This restriction is equivalent to a priori information. For QMIC and QMICC the a priori information is a vector $z \in R^n$, where the objective function value $f(z)$ is less than or equal to the optimal vlaue and for QMAC is a vector $z \in R^n$ in the convex feasible region. By translating the origin to z and setting $f'(x) = \max\{f(z), f(x+z)\}$, we can verify the restriction on the equivalent problems with the objective function f'. From a computational point of view, this transformation is practical if the task of obtaining the a priori information (vector z) is simpler than solving the original problem. This is true for most of QMAC and QMICC problems (see [24]). To get such a priori information for QMIC, we have to solve an unconstrained quasi-convex minimization, which should be easier than the original one. An example is given in §8. Several other examples can be the following.

Example 9.1.

$$\begin{aligned} &\max\{f_i(x_i), i = 1, \ldots, n\} \to \inf \\ &\text{s.t. } x \in X, \\ &0 \leq x \leq b, \end{aligned}$$

where $X \subseteq R^n, b \in R^n, b \geq 0$, and $f_i : R \mapsto R$ is a nondecreasing function. In this example $z = 0$, because

$$\max\{f_i(0), i = 1, \ldots, n\} \leq \max\{f_i(x_i), i = 1, \ldots, n\} \quad \forall x \geq 0.$$

Example 9.2.

$$\begin{aligned} &(\langle c, x\rangle / \langle d, x\rangle) \to \inf \\ &\text{s.t. } x \in X, \\ &a \leq x \leq b, \end{aligned}$$

where $X \subseteq R^n, a, b, c, d \in R^n, d \geq 0, b \geq a > 0$. This fractional programming problem becomes very simple if the constraint $x \in X$ is omitted. To solve $\min\{(\langle c, x\rangle / \langle d, x\rangle) : a \geq x \geq b\}$, we solve the following parametric problem

$$\begin{aligned} &\min\{\alpha : \min_{a \leq x \leq b} \langle c - \alpha d, x\rangle = 0\}, \\ \Leftrightarrow\ &\min\left\{\alpha : \sum_{i=1}^{n} (a_i \cdot \max\{c_i - \alpha d_i, 0\} + b_i \cdot \min\{c_i - \alpha d_i, 0\}) = 0\right\}. \end{aligned}$$

In QMAC (respectively, QMICC) we need the upper (respectively, lower) semicontinuity of the objective for the existence of an optimal solution. The objective function is quasi-subdifferentiable on the set of optimal solutions (Theorems 6.1 and 7.1) or, equivalently, the dual problems are solvable. This is not true in QMIC. Therefore, in QMIC we need additionally the upper semicontinuity for the quasi subdifferentiability. However, these semicontinuity properties can be relaxed. For other extensions we are referred to a separate work.

In the remainder we discuss the relationships between the results in [24] and [25] and the results presented in this paper. In [24] a symmetric dualization between QMAC and QMICC is presented. The dualization is obtained by using the quasi conjugation. The basic materials behind the relationships between the primal problems and their dual ones are the separation theorems. In principle, the duality can be developed to more general spaces, where the algebraic and topological structures are rich enough for the use of the separation theorems (e.g., [2], [31], [30], [7], and [4]). Since a convex program can be, of course, regarded as a particular case of QMAC, we can raise a natural question about the connection between the convex duality and the nonconvex duality. In the convex duality the Lagrangian functional approach plays a crucial role, and it is based on the minimax principles. In [25] an attempt has been made to show a connection between the convex duality and the nonconvex duality by using the minimax principles. In this paper we present a nonconvex duality for QMIC. This class is multi-extremal. However, the duality scheme is similar as in the convex problems. The duality relationship is of the type min = max (min = − min or max = − max) whereas the duality relationship in QMAC and QMICC is of the type min = − max (min = min or max = max). The optimality condition at a feasible vector z is of the type $0 \in A(z)$, where $A(z)$ is a convex set depending on z in the dual space, whereas the optimality criterions in QMAC and QMICC are of the type $B(z) \subseteq A(z)$, where $B(z), A(z)$ are convex sets in the dual space (see [8] and [26]). When $B(z)$ is reduced to $\{0\}$, the criterion $B(z) \subseteq A(z)$ becomes $0 \in A(z)$. Therefore, a duality in QMAC and QMICC can be regarded as a generalization of the convex-type duality. In applications we can use a convex-type duality and a nonconvex-type duality in a consistent scheme (see, for example, the application in §8).

In [27], [28], [32], and [33] other applications of nonconvex-type duality were discussed. Finally we observe that by a known result [13] a theoretical complexity of finding a local minimum is not much different from that of finding a global minimum in general nonconvex programs. In this paper we presented an additional argument by proving that the computational scheme in the class QMIC of multi-extremal problems is not much different from the computational scheme in convex minimization problems.

Acknowledgments. The author expresses his thanks to Prof. W. Oettli for helpful comments on the duality in general mathematical programs. The author also thanks the referee, who gave a correction of Theorem 2.2 and the proof of Theorem 4.2, which is much simpler than the original one.

REFERENCES

[1] M. Atteia and A. El Qortobi, *Quasiconvex duality*, in Optimization and Optimal Control, Lecture Notes in Control and Information Science 30, Springer-Verlag, Berlin, 1981, pp. 3–8.
[2] R. E. Burkard, H. W. Hamacher, and J. Tind, *On abstract duality in mathematical programming*, Z. Oper. Res., 26 (1982), pp. 197–209.
[3] J. P. Crouzeix, *Duality framework in quasiconvex programming*, in Generalized Concavity in Optimization and Economics, S. Schaible and W. T. Ziemba, eds., Academic Press, New York, 1981, pp. 207–226.
[4] K. Deimling, *Nonlinear Functional Analysis*, Springer-Verlag, Berlin, 1985.
[5] H. P. Greenberg and W. P. Pierskalla, *Quasiconjugate functions and surrogate duality*, Cahiers Centre Etudes Rech. Opér., 15 (1973), pp. 437–448.
[6] M. Grotschel, L. Lovasz, and A. Schrijver, *Geometric Algorithms and Combinatorial Optimization*, Springer-Verlag, Berlin, 1988.
[7] J.-B. Hiriart-Urruty, *Generalized differentiability, duality and optimization for problems dealing with differences of convex functions*, Lecture Notes in Econom. Math. Systems 256, J. Ponstains, ed., Springer-Verlag, New York, 1984, pp. 37–70.
[8] ———, *Conditions nécessaires et suffisantes d'optimalité global en optimisation de deux fonctions convexes*, C. R. Acad. Sci. Paris Sér., 1, 1989, pp. 459–462.
[9] R. Horst, *On the convexification of nonlinear programming problems: An application-oriented survey*, European J. Oper. Res., 15 (1984), pp. 382–392.
[10] R. Horst and H. Tuy, *Global Optimization*, Springer-Verlag, New York, 1990.
[11] L. Lovasz, *Geometric algorithms and algorithmic geometry*, Plenary Lecture at the International Congress of Mathematicians, Kyoto, Japan, 1990.
[12] J. E. Martinez-Legaz, *Quasiconvex duality theory by generalized conjugation methods*, Optimization, 19 (1988), pp. 603–652.
[13] K. G. Murty and S. H. Kabadi, *Some NP-complete problems in quadratic and nonlinear programming*, Math. Programming, 39 (1987), pp. 117–129.
[14] W. Oettli, *Optimality conditions involving generalized convex mappings*, in Generalized Concavity in Optimization and Economics, S. Schailble and W. T. Ziemba, eds., Academic Press, New York, 1981, pp. 227–238.
[15] ———, *Optimality conditions for programming problems involving multivalued mapping*, in Applied Mathematics, B. Korte, ed., North-Holland, Amsterdam, 1982, pp. 196–226.
[16] U. Passy and E. Z. Prisman, *A convex-like duality scheme for quasiconvex programs*, Math. Programming, 32 (1985), pp. 278–300.
[17] J. P. Penot and M. Volle, *On quasi-convex duality*, Math. Oper. Res., 15 (1990), pp. 597–625.
[18] E. Polak, *An implementable algorithm for the optimal design centering, tolerancing and tuning problem*, J. Optim. Theory Appl., 37 (1982), pp. 45–67.
[19] B. N. Pshenichnyyi, *Leçons sur jeux differentiels, control optimal et jeux differentiels*, Cahiers de IIRIA, 4 1971.
[20] T. R. Rockafellar, *Convex Analysis*, Princeton University Press, Princeton, NJ, 1970.
[21] I. Singer, *Surrogate functionals and surrogate convexity*, Appl. Anal., 16 (1983), pp. 291–327.
[22] ———, *A general theory of dual optimization problems*, J. Math. Anal. Appl., 116 (1986), pp. 77–130.
[23] P. T. Thach, *The design centering problem as a d.c. programming problem*, Math. Programming, 41 (1988), pp. 229–248.
[24] ———, *Quasiconjugates of functions, duality relationship between quasiconvex minimization under a reverse convex constraint and quasiconvex maximization under a convex constraint, and applications*, J. Math. Anal. Appl., 159 (1991), pp. 299–322.

[25] P. T. Thach, *A generalized duality and applications*, J. Global Optimization, 3 (1993), pp. 311–324.
[26] ———, *Global optimality criterions and a duality with a zero gap in nonconvex optimization problems*, SIAM J. Math. Anal., 24 (1993), pp. 1537–1556.
[27] P. T. Thach, R. E. Burkard, and W. Oettli, *Mathematical programs with a two-dimensional reverse convex constraint*, J. Global Optim., 1 (1991), pp. 145–154.
[28] P. T. Thach and H. Tuy, *Dual outer approximation methods for concave programs and reverse convex programs*, Report 90-30, Institute of Human and Social Sciences, Tokyo Institute of Technology, 1990.
[29] J. Tind, *Blocking and antiblocking sets*, Math. Programming, 6 (1974), pp. 157–166.
[30] J. Tind and L. A. Wolsey, *An elementary survey of general duality theory in mathematical programming*, Math. Programming, 21 (1981), pp. 241–261.
[31] J. F. Toland, *Duality in nonconvex optimization*, J. Math. Anal. Appl., 66 (1978), pp. 399–415.
[32] H. Tuy, *A general deterministic approach to global optimization via d.c. programming*, Math. Studies, 129 (1987), pp. 273–303.
[33] ———, *Polyhedral annexation, dualization and dimension reduction technique in global optimization*, J. Global Optim., 1 (1991), pp. 229–244.

SIAM J. OPTIMIZATION
Vol. 4, No. 1, pp. 65–85, February 1994
©1994 Society for Industrial and Applied Mathematics
003

A PATH-FOLLOWING PROJECTIVE INTERIOR POINT METHOD FOR LINEAR PROGRAMMING*

DONG X. SHAW† AND DONALD GOLDFARB‡

Abstract. In this paper the authors present a projective interior point method which follows the "central trajectory" and finds an optimal solution in at most $O(\sqrt{n}L)$ iterations. This algorithm is essentially Anstreicher's variant of Karmarkar's projective algorithm with a weakened "ball update" for lower bounds. The result indicates that a strong relationship exists between projective and path-following algorithms. The authors show that within a certain probabilistic framework the expected number of iterations required by their algorithm is $O(L)$.

Key words. projective interior point method, path-following method, central trajectory, polynomial-time complexity, expected number of iterations

AMS subject classifications. 49D, 90C05

1. Introduction.

The primary objective of this paper is to investigate the nature and worst-case behavior of projective interior point methods to shed light on the relationship that exists between these methods and path-following methods. In this regard, we present a new projective interior point method for linear programming that is identical to Anstreicher's [2] lower bound variant of Karmarkar's [13] projective algorithm except that a larger "ball" is used to update the lower bound on the optimal objective function value. We show that this algorithm is "path-following," and hence, not surprisingly, that its step complexity is $O(\sqrt{n}L)$, rather than $O(nL)$, where n is the number of variables and L is the input length.

Many projective interior point methods have been proposed since the publication by Karmarkar [13] of his original projective algorithm. These include the lower bound variants of Anstreicher [2], de Ghellinck and Vial [6], and Todd and Burrell [24], and the primal projective method of Goldfarb and Shaw (Xiao) [7], all of which have worst-case step-complexities of $O(nL)$.

Path-following algorithms, on the other hand, have, in general, a worst-case step complexity of $O(\sqrt{n}L)$. (See, e.g., [9], [15], [17], and [20].) These algorithms all generate points that closely follow the central trajectory, and none, as far as we know, use projective transformations. We note that "large-step" versions of path-following methods have been developed [10], [22]. These versions require the solution path to be close to the central trajectory only at certain points along it and, in general, have either $O(nL)$ or $O(\sqrt{n}L)$ step-complexities, depending on how aggressively a barrier parameter is decreased.

All path-following algorithms that have been proposed, with the exception of the algorithm in this paper, were developed by first defining the central trajectory as the path of minima of a logarithmic barrier function or a potential function, con-

*Received by the editors May 23, 1990; accepted for publication (in revised form) November 3, 1992. This research was supported in part by Office of Naval Research contract N-00014-87-K0214, National Science Foundation grants DMS-85-12277 and DMS-91-06195, and Department of Energy grant DE-FG02-92ER25126.

†School of Industrial Engineering, Purdue University, West Lafayette, Indiana 47907 (shawdx@ecn.purdue.edu).

‡Department of Industrial Engineering and Operations Research, Columbia University, New York, New York 10027 (goldfarb@cunixb.columbia.edu).

strained by linear equalities if a standard form linear program is being solved. Both approaches define the same trajectory. In the former case, the minima (i.e., points on the trajectory), are parametrized by the barrier parameter as in the methods of Gonzaga [9], Monteiro and Adler [17], and Roos and Vial [20], while in the latter case the parameter is either a lower bound on the objective value, as in Roos's method [20], or an upper bound, as in Renegar's method [19]. (Here we are interpreting the results in [19] and [20] on maximization problems in terms of minimization problems.) Given a point close to a particular point on the central trajectory, a typical short-step path-following algorithm takes a single Newton step to get closer to this point, and then updates the parameter by a small amount.

As we shall show, our slightly modified version of Anstreicher's algorithm can be viewed as operating in a manner similar to the above path-following algorithms, with one exception: the update of the parameter (lower bound) in our algorithm is obtained implicitly by solving a quadratic equation rather than by adding or subtracting an explicit amount to the parameter as in the other algorithms. However, we would like to emphasize that our projective algorithm, and certainly Anstreicher's original projective algorithm, were not derived using a path-following framework. Consequently, the result that our projective algorithm is path-following is not only interesting, it is rather surprising.

After deriving basic results about the ball minimization problem and the lower bound computation in the next section, we present in §3 a projective interior point algorithm which is similar to Anstreicher's lower bound variant of Karmarkar's algorithm. We assume that the initial point is close to the "central trajectory," and prove that the algorithm is "path-following" and has a worst-case iteration bound of $O(\sqrt{n}L)$. A procedure for obtaining an interior point that satisfies the initial conditions required by our algorithm is described in §4. A probabilistic model for explaining the practical behavior of projective algorithms is described in §5. Finally, in an appendix we give a proof that the problem obtained by replacing the objective function of a linear program by the lower bound version of Karmarkar's potential function has a unique solution satisfying the first-order optimality conditions.

2. The lower bound computation and the ball minimization problem. Throughout this paper we consider the linear programming problem

$$(P) \qquad \begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax = 0, \\ & x_n = 1, \\ & x \geq 0, \qquad x \in R^n, \end{aligned}$$

and its dual,

$$(D) \qquad \begin{aligned} \max \quad & v \\ \text{s.t.} \quad & \eta = c - A^T\pi - ve_n \geq 0, \end{aligned}$$

where e_n is the nth unit coordinate vector. We assume that both (P) and (D) have interior feasible solutions, the matrix A has full row rank, and $n \geq 3$. We also assume that the input data are all integer and the total length (i.e., the number of digits) of the input data is L.

Given a nonoptimal feasible interior point $x^k > 0$ for the primal problem (P), define the *projective transformation*:

$$(2.1) \qquad T(x): \quad x \mapsto y = \frac{nD^{-1}x}{e^T D^{-1}x},$$

which maps $\{x \in R^n \mid x \geq 0,\ x_n = 1\}$ onto $S^n \equiv \{y \in R^n \mid y \geq 0,\ e^T y = n\}$, where $D = \operatorname{diag}\{x^k\}$ and e is a vector of all ones. Since $x_n^k = x_n = 1$, $y_n = n/(e^T D^{-1} x)$ and the inverse transformation is

$$T^{-1}(y): \quad y \mapsto x = \frac{Dy}{y_n}.$$

Clearly, $e = T(x^k)$. Also, let us introduce the dual transformation,

$$s = D\eta,$$

and denote $s^k = D\eta^k$. Under these transformations, the original primal-dual pair of linear programs become

$$(\hat{P}_k) \qquad \begin{aligned} \min \quad & f(y) = \frac{\hat{c}^T y}{y_n} \\ \text{s.t.} \quad & \hat{A}y = 0, \\ & e^T y = n, \\ & y \geq 0, \end{aligned}$$

and

$$(\hat{D}_k) \qquad \begin{aligned} \max \quad & v \\ \text{s.t.} \quad & s = \hat{c} - \hat{A}^T \pi - v e_n \geq 0, \end{aligned}$$

where $\hat{c} = Dc$ and $\hat{A} = AD$.

Now consider the *ball minimization problem*:

$$(B_\rho) \qquad \begin{aligned} \min \quad & f(y) = \frac{\hat{c}^T y}{y_n} \\ \text{s.t.} \quad & \hat{A}y = 0, \\ & e^T y = n, \\ & \|y - e\| \leq \rho, \qquad y \in R^n. \end{aligned}$$

Notice that (B_ρ) is a relaxation of $(\hat{P}_k)$ if $\rho \geq R$ and it is a restriction of $(\hat{P}_k)$ if $\rho \leq r$, where

$$r = \sqrt{\frac{n}{n-1}} \quad \text{and} \quad R = \sqrt{n(n-1)}$$

are, respectively, the radii of the largest and smallest balls in R^n that inscribe and circumscribe the simplex S^n.

This ball minimization problem with $\rho = R$ is used in Anstreicher's [2] projective algorithm to compute lower bounds on the optimal objective value of problem (P). In the next section we show that by using a ball of radius $R_1 > R$ that *strictly* circumscribes the simplex S^n, one obtains a projective algorithm that is path-following and has an improved worst-case step-complexity $O(\sqrt{n}L)$.

The level sets of the fractional linear objective function in problem (B_ρ) are the affine sets

$$H(v) = \{y \in R^n \mid (\hat{c} - v e_n)^T y = 0, \hat{A}y = 0, e^T y = n\}.$$

Let $c_p = P\hat{c}, d_p = Pe_n$, where P is the orthogonal projection operator from R^n to

$$\{y \in R^n |\ \hat{A}y = 0,\ e^T y = 0\}.$$

If c_p and d_p are not collinear, all the level sets $\{H(v), v \in R^1\}$ intersect in an affine set M, which we shall call the *singular set* (see [7]), since $\hat{c}^T y = y_n = 0$, for $y \in M$. The intersection of M with the two-dimensional space spanned by c_p and d_p is the *singular point* $e + w$, where

$$w = \frac{(\hat{c}_p^T d_p)c_p - (\hat{c}_p^T c_p)d_p}{\|c_p\|^2\|d_p\|^2 - (c_p^T d_p)^2}, \tag{2.2}$$

$f^k = \hat{c}^T e = c^T De = c^T x^k$, and $\hat{c}_p = c_p - f^k d_p$. A proof of this fact and other properties of w can be found in [7] and [23]. Also, it is easy to verify that

$$\|w\|^2 = \frac{\|c_p - f^k d_p\|^2}{\|c_p\|^2\|d_p\|^2 - (c_p^T d_p)^2}, \tag{2.3}$$

and hence that $\|w\| = \infty$ if $c_p = \nu d_p$ with $\nu \neq f^k$. Our assumption that x^k is a nonoptimal interior point, or, equivalently, that the objective function is not constant over the feasible region, ensures that $c_p \neq f^k d_p$, and hence that $\|w\|$ is well defined. The solution of problem (B_ρ) is closely related to the following equation:

$$E_\rho(v): \qquad \|c_p - v d_p\|\rho = f^k - v,$$

whose derivation can be found in [7] and [23]. This equation can be written as the quadratic equation

$$\|c_p - v d_p\|^2\rho^2 = (f^k - v)^2.$$

Letting $z = f^k - v$, we get a quadratic equation in z,

$$Q_\rho(z): \qquad \|c_p - f^k d_p + z d_p\|^2\rho^2 = z^2, \tag{2.4}$$

which has two roots:

$$z^{\pm}(\rho) = \frac{(c_p - f^k d_p)^T d_p \pm \sqrt{\Delta(\rho)}}{\dfrac{1}{\rho^2} - \|d_p\|^2} \tag{2.5}$$

$$= \frac{\|c_p - f^k d_p\|^2}{\pm\sqrt{\Delta(\rho)} - (c_p - f^k d_p)^T d_p}, \tag{2.6}$$

where

$$\Delta(\rho) = \{(c_p - f^k d_p)^T d_p\}^2 + \|c_p - f^k d_p\|^2 \left(\frac{1}{\rho^2} - \|d_p\|^2\right). \tag{2.7}$$

Obviously, any root of $E_\rho(v)$ can be converted to a root of $Q_\rho(z)$; conversely, any positive root of $Q_\rho(z)$ can be converted to a root of $E_\rho(v)$.

Let $v^k \leq f^*$ be a given lower bound for the optimal objective value f^* of problem (P), and suppose that the current point x^k is not optimal. The level set corresponding to v^k is

$$\begin{aligned} H(v^k) &= \{y \in R^n \mid (\hat{c} - v^k e_n)^T y = 0, \hat{A}y = 0, e^T y = n\} \\ &= \{y \in R^n \mid (c_p - v^k d_p)^T (y - e) = -(f^k - v^k),\ \hat{A}y = 0, e^T y = n\}. \end{aligned} \tag{2.8}$$

If $H(v^k) \neq \phi$, (2.8) and $f^k \neq v^k$ together imply that $c_p - v^k d_p \neq 0$; hence, we can define

$$\rho^k = \frac{f^k - v^k}{\|c_p - v^k d_p\|} \tag{2.9}$$

and

$$\tilde{y}^k = e - \rho^k \frac{c_p - v^k d_p}{\|c_p - v^k d_p\|}. \tag{2.10}$$

If $H(v^k) = \phi$, then $c_p - v^k d_p = 0$, and we technically define $\rho^k = \infty$.

For $H(v^k) \neq \phi$, it is easy to verify that v^k is a root of $E_{\rho^k}(v)$ and $\tilde{y}^k \in H(v^k)$. Geometrically, ρ^k is the radius of the ball with center e that is tangent to the level set $H(v^k)$. This fact can be verified directly from the expression for $H(v^k)$ in (2.8). Also, the geometric meaning of equation $E_\rho(v)$ is that its roots determine the level sets that are tangent to the ball with radius ρ and center e. (See also [23, Chap. 2].)

LEMMA 2.1. *Assume that f^k is not the optimal value, and v^k is a lower bound with $H(v^k) \neq \phi$.*

(i) *For any $y \in H(v^k)$, $\|y - e\| \geq \rho^k$.*

(ii) *$\rho^k > \rho$ implies $\|w\| > \rho$, i.e., the singular point is outside of the ball with radius ρ and center e.*

(iii) *$Q_\rho(v)$ is solvable and has two real roots if and only if $\|w\| > \rho$.*

(iv) *Assume v' is a solution of*

$$E_\rho(v): \qquad \|c_p - v d_p\| \rho = f^k - v,$$

and $z^\pm(\rho)$ are the two roots of $Q_\rho(z)$. Then,

$$(c_p - f^k d_p)^T (c_p - v' d_p) = \pm \sqrt{\Delta(\rho)} z^\pm(\rho),$$

where $z^\pm(\rho) > 0$, and the $\pm$ notation means $+$ if $f^k - v' = z^+(\rho)$ and $-$ if $f^k - v' = z^-(\rho)$.

Proof. (i) Since by (2.8), $\tilde{y}^k - e$ is orthogonal to the level set $H(v^k)$ and $\tilde{y}^k \in H(v^k)$,

$$\|y - e\| \geq \|\tilde{y}^k - e\| = \rho^k \quad \text{for all } y \in H(v^k).$$

(ii) Since the singular point $e + w \in H(v^k)$,

$$\|w\| \geq \|\tilde{y}^k - e\| = \rho^k > \rho.$$

(iii) By (2.7), $\Delta(\rho) > 0$ is equivalent to

$$\begin{aligned} \|c_p - f^k d_p\|^2 / \rho^2 &> \|c_p - f^k d_p\|^2 \|d_p\|^2 - \{(c_p - f^k d_p)^T d_p\}^2 \\ &= \|c_p\|^2 \|d_p\|^2 - (c_p^T d_p)^2. \end{aligned}$$

Moreover, $c_p - f^k d_p \neq 0$ since $f^k > f^*$. Hence, from (2.3), $\|w\|^2 > \rho^2$ if and only if $\Delta(\rho) > 0$, or equivalently, if and only if $Q_\rho(z)$ is solvable and has two real roots.

(iv) From $f^k - v' = z^\pm(\rho)$ and (2.6), we obtain

$$\begin{aligned} (c_p - f^k d_p)^T (c_p - v' d_p) &= \|c_p - f^k d_p\|^2 + (f^k - v')(c_p - f^k d_p)^T d_p \\ &= \pm \sqrt{\Delta(\rho)} z^\pm(\rho). \end{aligned}$$

□

We now consider balls in R^n of radii $r_1 = \alpha r$ and $R_1 = R/\alpha$, which, respectively, strictly inscribe and circumscribe the simplex S^n, where α is a positive constant that satisfies $r_1 \leq \frac{2}{3}$.

LEMMA 2.2. *Assume that the primal feasible solution $x^k > 0$ is not optimal. Let v^k be a lower bound for (P). If $\rho^k > R_1$, then we have the following results.*

(i) *The quadratic equation $Q_{R_1}(z)$ is solvable, and $z^+(R_1) > 0$.*

(ii) *$\underline{v} \equiv f^k - z^+(R_1)$ provides a lower bound for (P) and $\underline{v} > v^k$. Moreover, there exists $\underline{s}$, an interior dual feasible solution with dual objective value $\underline{v}$, such that*

$$\underline{s} = \frac{f^k - \underline{v}}{n}(e - \Delta y) > 0,$$

where

$$\Delta y = -r_1 \frac{c_p - \underline{v} d_p}{\|c_p - \underline{v} d_p\|}. \tag{2.11}$$

(iii) *Δy is a descent direction for $(\hat{P}_k)$ at e.*

Proof. (i) It directly follows from Lemma 2.1 (ii) and (iii) with $\rho = R_1$ that $Q_{R_1}(z)$ is solvable. To show that $z^+(R_1) > 0$ just notice that if $(c_p - f^k d_p)^T d_p \leq 0$, we immediately have $z^+(R_1) > 0$ by (2.6). Otherwise, notice that $(c_p - f^k d_p)^T d_p > 0$ implies that $(1/(\rho^k)^2) - \|d_p\|^2 \geq 0$. (If not, then by (2.6) and (2.7), both $z^+(\rho^k)$ and $z^-(\rho^k)$ are negative. But by the definition (2.9) of ρ^k, v^k is a root of $E_{\rho^k}(v)$, so $f^k - v^k$ is a positive root of $Q_{\rho^k}(z)$—a contradiction.) Hence,

$$\frac{1}{R_1^2} - \|d_p\|^2 > \frac{1}{(\rho^k)^2} - \|d_p\|^2 \geq 0,$$

and it follows from (2.5) that $z^+(R_1) > 0$.

(ii) Define $\underline{v} = f^k - z^+(R_1)$, and let

$$\hat{y} = e - \frac{c_p - \underline{v} d_p}{\|c_p - \underline{v} d_p\|} R_1. \tag{2.12}$$

Then, by equation $E_{R_1}(\underline{v})$ and (2.12), it follows that

$$\begin{aligned} c_p - \underline{v} d_p &= \|c_p - \underline{v} d_p\| \frac{e - \hat{y}}{R_1} \\ &= (f^k - \underline{v}) \frac{e - \hat{y}}{R_1^2}. \end{aligned} \tag{2.13}$$

On the other hand,

$$\begin{aligned} c_p - \underline{v} d_p = P(\hat{c} - \underline{v} e_n) &= P_e P_{\hat{A}}(\hat{c} - \underline{v} e_n) \\ &= \hat{c} - \underline{v} e_n - \hat{A}^T \pi' - \frac{1}{n} e^T (\hat{c} - \underline{v} e_n) e \\ &= \hat{c} - \underline{v} e_n - \hat{A}^T \pi' - \frac{1}{n}(f^k - \underline{v}) e. \end{aligned} \tag{2.14}$$

Together with (2.13), we have

$$\begin{aligned} \underline{s} \equiv \hat{c} - \underline{v} e_n - \hat{A}^T \pi' &= \frac{1}{n}(f^k - \underline{v}) e + (f^k - \underline{v}) \frac{e - \hat{y}}{R_1^2} \\ &= \frac{1}{n}(f^k - \underline{v}) \left(e - \frac{n}{R_1^2}(\hat{y} - e) \right) \\ &= \frac{1}{n}(f^k - \underline{v})(e - \Delta y), \end{aligned}$$

where Δy is defined by (2.11). Since $\|\Delta y\| = r_1 < 1$ and $f^k - \underline{v} = z^+(R_1) > 0$, $\underline{s} > 0$, and consequently, $\underline{v}$, is a lower bound. Moreover, since $\rho^k > R_1$, it follows from (2.7) that $\Delta(\rho^k) < \Delta(R_1)$, and hence from (2.6) that $z^+(\rho^k) > z^+(R_1)$. Therefore, $\underline{v} = f^k - z^+(R_1) > f^k - z^+(\rho^k) = v^k$.

To prove (iii), observe that $\underline{v}$ is a solution of $E_{R_1}(v)$, and $f^k - \underline{v} = z^+(R_1)$. Therefore, by Lemma 2.1 (iv),

$$(c_p - f^k d_p)^T (c_p - \underline{v} d_p) = \sqrt{\Delta(R_1)} z^+(R_1) > 0. \qquad \square$$

3. A path-following projective algorithm. Before presenting our algorithm, we first define the "central trajectory" or "central path" for problem (P). There are several ways to do this (see [4], [16], [19], and [21]). One way is obtained by replacing the linear objective function in (P) by the logarithmic barrier function, which yields the family of problems:

$$(BP_\mu) \qquad \begin{aligned} \min \quad & B(x,\mu) = c^T x - \mu \sum_{j=1}^{n} \ln x_j \\ \text{s.t.} \quad & Ax = 0, \\ & x_n = 1, \\ & x > 0, \qquad x \in R^n. \end{aligned}$$

For any given $0 < \mu < \infty$, problem (BP_μ) has a unique optimal solution $x = x^1(\mu)$, which satisfies the constraints of problem (BP_μ) and

$$x * \eta - \mu e = 0^{\P}, \tag{3.1}$$

where

$$\eta = c - A^T \pi - v e_n > 0 \tag{3.2}$$

and

$$v = c^T x - n\mu. \tag{3.3}$$

Here, v is the dual objective value of the dual feasible solution η. (See [16] for details.) On the other hand, if (3.1)–(3.3) hold for some $\mu > 0$ and x satisfies the constraints of (BP_μ), then x is the unique optimal solution of (BP_μ). The central trajectory can then be described as the set of solutions $\{x \mid x = x^1(\mu),\ 0 < \mu < \infty\}$ of (3.1)–(3.3) and the constraints of (BP_μ).

Another way to define the "central trajectory" is to use the lower bound version of Karmarkar's potential function with the lower bound v on the optimal objective value f^* of problem (P) as a path parameter. Formally, consider the following family of problems (PP_v) for $-\infty < v < f^*$:

$$(PP_v) \qquad \begin{aligned} \min \quad & p(x,v) = n \ln(c^T x - v) - \sum_{j=1}^{n} \ln x_j \\ \text{s.t.} \quad & Ax = 0, \\ & x_n = 1, \\ & x > 0, \qquad x \in R^n. \end{aligned}$$

¶The operation " $*$ " from $R^n \times R^n \to R^n$ is defined as: $x * y = z$, where $z = (x_1 y_1, \ldots, x_n y_n)$, and is known as the *Hadamard product.*

Although the objective function of (PP_v) is nonconvex, we show in an appendix that a point satisfies the Karush–Kuhn–Tucker conditions for (PP_v) if and only if it is the unique optimal solution $x = x^2(v)$ of problem (PP_v). From these conditions, we have

$$c - \mu D^{-1}e - A^T\pi - ve_n = 0, \tag{3.4}$$

where $D = \operatorname{diag}\{x\}$ and

$$\mu = \frac{c^T x - v}{n}. \tag{3.5}$$

Clearly, the set of solutions $\{x|\ x = x^2(v),\ -\infty < v < f^*\}$ to (3.4) and (3.5) is identical to the set of solutions $\{x|\ x = x^1(\mu),\ 0 < \mu < \infty\}$ to (3.1)–(3.3), and defines the same "central trajectory." The one-to-one correspondence between $x^1(\mu)$ and $x^2(v)$ is given by (3.3) and (3.5), and was first pointed out by Gonzaga [10].

Moreover, if $v < f^*$, then it follows from (2.14) that the conditions (3.1)–(3.3) or (3.4)–(3.5) for a feasible point x^k to be the point $x^1(\mu)$ or $x^2(v)$ on the central path are satisfied if and only if $c_p - vd_p = 0$. In this case c_p and d_p are collinear, $\|w\| = \infty$, and there is no singular point.

We are now ready to present a projective algorithm for solving the linear program (P) that is path-following, and hence, not surprisingly, has a worst-case step-complexity of $O(\sqrt{n}L)$ rather than $O(nL)$. This algorithm is essentially identical to Anstreicher's [2] lower bound variant of Karmarkar's algorithm except that it uses the "weakened ball" update of the lower bound described in Lemma 2.2 and starts from a point close to the "central trajectory."

ALGORITHM 1.

1° *Given $x^0 > 0$, a nonoptimal primal feasible solution with primal objective value f^0, and a lower bound v^0 (associated with an interior dual feasible solution), satisfying $\rho^0 \equiv (f^0 - v^0)/\|c_p - v^0 d_p\| > R_1$, where $r_1 = \alpha R$ and $R_1 = R/\alpha$, and $\alpha > 0$ is chosen so that $r_1 \le 2/3$. Set $k \leftarrow 0$.*

2° *While $f^k - v^k \equiv c^T x^k - v^k > \varepsilon$, $\quad (\ \varepsilon = 2^{-L})$*
solve $Q_{R_1}(z)$ to obtain $z^+(R_1)$, and set

$$v^{k+1} \leftarrow \underline{v} = f^k - z^+(R_1),$$

$$\Delta y \leftarrow -r_1 \frac{c_p - v^{k+1} d_p}{\|c_p - v^{k+1} d_p\|}.$$

Then, set

$$y^{k+1} \leftarrow e + \Delta y,$$

$$f^{k+1} \leftarrow f(y^{k+1}),$$

$$x^{k+1} \leftarrow T^{-1}(y^{k+1}),$$

$$k \leftarrow k + 1.$$

3° *Round x^k to a vertex x^* while keeping the objective function nonincreasing. Output x^* as an optimal solution.*

An arbitrary feasible point x is said to be close to the central trajectory (e.g., see [17]) if there exists a $\mu > 0$ (or, equivalently, a $v < f^*$) and a dual feasible solution η satisfying (3.2) and (3.3) and

$$\|x * \eta - \mu e\| \leq \theta\mu \tag{3.6}$$

for some fixed $\theta < 1$. Theorem 1 shows that the condition $\rho^k > R_1$, where ρ^k is defined by (2.9), is equivalent to the closeness condition (3.6) with $\theta = r_1$, and that this condition is satisfied at every iteration of Algorithm 1. (Note that by Lemma 2.2 this also establishes that Algorithm 1 is well defined.) Hence, Algorithm 1 is a path-following algorithm and $(\rho^k)^{-1}$ can be viewed as a measure of the proximity of the current iterate x^k to the point $x^2(v^k)$ on the central path. Note that the condition $\rho^k > R_1$ is equivalent to the bound

$$\|c_p - v^k d_p\| < (c^T x^k - v^k)/R_1$$

on the norm of the projected Newton direction, and that this condition is related to conditions used in other interior point algorithms to measure the closeness of the current iterate to the central path (e.g., see [22]).

THEOREM 1. *Let $\{x^k\}$ be a sequence of iterative points generated by Algorithm 1.*

(i) *$\rho^k > R_1$ if and only if there exists η^k, a dual feasible solution of (D), with dual objective value v^k such that*

$$\|x^k * \eta^k - \mu^k e\| < \mu^k r_1, \tag{3.7}$$

where $\mu^k = (f^k - v^k)/n$.

(ii) *If at iteration k, $\rho^k > R_1$, then at iteration $k+1$,*

$$\rho^{k+1} > R_2 > R_1,$$

where $R_2 = R/(\alpha r_1) = R_1/r_1 > R_1$.

Proof. (i) To prove necessity, notice that from (2.9) and (2.10)

$$\begin{aligned} c_p - v^k d_p &= \|c_p - v^k d_p\| \rho^k \frac{e - \tilde{y}^k}{(\rho^k)^2} \\ &= (f^k - v^k)\frac{e - \tilde{y}^k}{(\rho^k)^2} \\ &\equiv \mu^k g, \end{aligned} \tag{3.8}$$

where $g = (n/(\rho^k)^2)(e - \tilde{y}^k)$. Also, as in (2.14),

$$\begin{aligned} c_p - v^k d_p &= \hat{c} - v^k e_n - \hat{A}^T \pi' - \frac{(f^k - v^k)}{n} e \\ &= \hat{c} - v^k e_n - \hat{A}^T \pi' - \mu^k e. \end{aligned} \tag{3.9}$$

Hence, by (3.8) and (3.9)

$$s^k \equiv \hat{c} - v^k e_n - \hat{A}^T \pi' = \mu^k (e + g).$$

Since $\|g\| = n/\rho^k < n/R_1 = r_1 < 1$, we have $s^k > 0$, and

$$\|s^k - \mu^k e\| = \mu^k \|g\| < \mu^k r_1. \tag{3.10}$$

Obviously, the dual objective value corresponding to s^k is v^k. By the dual transformation, the proof is complete.

To prove the sufficiency, let η^k be a dual feasible solution of (D) with dual objective value v^k such that (3.7) is satisfied. Then

$$s^k \equiv x^k * \eta^k = \hat{c} - v^k e_n - \hat{A}^T \pi''$$

for some π''. From (3.9),

$$\begin{aligned} c_p - v^k d_p &= \hat{c} - v^k e_n - \hat{A}^T \pi' - \mu^k e \\ &= s^k - \hat{A}^T \Delta\pi - \mu^k e, \end{aligned}$$

where $\Delta\pi = \pi' - \pi''$. So,

$$c_p - v^k d_p + \hat{A}^T \Delta\pi = s^k - \mu^k e. \tag{3.11}$$

Hence,

$$\begin{aligned} \|c_p - v^k d_p\|^2 + \|\hat{A}^T \Delta\pi\|^2 &= \|s^k - \mu^k e\|^2 \\ &= \|x^k * \eta^k - \mu^k e\|^2 < (\mu^k r_1)^2. \end{aligned}$$

Therefore,

$$\|c_p - v^k d_p\| R_1 < \mu^k r_1 R_1 = n\mu^k = f^k - v^k. \tag{3.12}$$

(ii) By the algorithm, $v^{k+1} = \underline{v} = f^k - z^+(R_1)$, and from Lemma 2.2,

$$s' \equiv \hat{c} - v^{k+1} e_n - \hat{A}^T \pi' = \frac{f^k - v^{k+1}}{n}(e - \Delta y) > 0. \tag{3.13}$$

Hence, as $(y^{k+1})^T \hat{A}^T = 0$,

$$\begin{aligned} (\hat{c} - v^{k+1} e_n)^T y^{k+1} &= \frac{f^k - v^{k+1}}{n}(y^{k+1})^T (e - \Delta y) \\ &= \frac{f^k - v^{k+1}}{n}(e + \Delta y)^T (e - \Delta y) \\ &= \frac{f^k - v^{k+1}}{n}(n - \|\Delta y\|^2) \\ &= (f^k - v^{k+1})\left(1 - \frac{r_1^2}{n}\right), \end{aligned}$$

or

$$\frac{(f^{k+1} - v^{k+1}) y_n^{k+1}}{f^k - v^{k+1}} = \frac{(\hat{c} - v^{k+1} e_n)^T y^{k+1}}{f^k - v^{k+1}} = 1 - \frac{r_1^2}{n}. \tag{3.14}$$

Note that $\hat{c} = Dc$, $\hat{A}^T = DA^T$, and $x_n = 1$, where $D \equiv \operatorname{diag}\{x^k\}$. It follows immediately from (3.13) that

$$\eta' = c - v^{k+1} e_n - A\pi'$$

is a dual feasible solution with dual objective value v^{k+1}. Since

$$x^{k+1} = \frac{Dy^{k+1}}{y_n^{k+1}} = x^k * \left(\frac{y^{k+1}}{y_n^{k+1}}\right) \quad \text{and} \quad y^{k+1} = e + \Delta y,$$

it follows from (3.13) and (3.14) that

$$\begin{aligned} x^{k+1} * \eta' &= \left(\frac{y^{k+1}}{y_n^{k+1}}\right) * s' \\ &= \frac{f^k - v^{k+1}}{n}\left(\frac{y^{k+1}}{y_n^{k+1}}\right) * (e - \Delta y) \\ &= \frac{f^k - v^{k+1}}{n y_n^{k+1}}(e + \Delta y) * (e - \Delta y) \\ &= \frac{f^{k+1} - v^{k+1}}{n}\left(1 - \frac{r_1^2}{n}\right)^{-1}(e - \Delta y * \Delta y). \end{aligned} \tag{3.15}$$

Letting $\mu^{k+1} = \frac{1}{n}(f^{k+1} - v^{k+1}) > 0$, it follows from (3.15) that

$$\begin{aligned} \|x^{k+1} * \eta' - \mu^{k+1}e\|^2 &= (\mu^{k+1})^2\left(1 - \frac{r_1^2}{n}\right)^{-2}\left\|\frac{r_1^2}{n}e - \Delta y * \Delta y\right\|^2 \\ &= (\mu^{k+1})^2\left(1 - \frac{r_1^2}{n}\right)^{-2}\left[\frac{r_1^4}{n} + \sum_{j=1}^n \Delta y_j^4 - 2\frac{r_1^2}{n}\sum_{j=1}^n \Delta y_j^2\right] \\ &\le (\mu^{k+1})^2\left(1 - \frac{r_1^2}{n}\right)^{-2}\left[\frac{r_1^4}{n} + r_1^4 - 2\frac{r_1^4}{n}\right] \\ &= (\mu^{k+1})^2\frac{n^2 - n}{(n - r_1^2)^2}r_1^4 < (\mu^{k+1})^2 r_1^4 \qquad \left(n \ge 3, 0 < r_1 \le \frac{2}{3}\right), \end{aligned}$$

where the first inequality follows from $\sum_{j=1}^n \Delta y_j^4 \le (\sum_{j=1}^n \Delta y_j^2)^2 = r_1^4$. Hence,

$$\|x^{k+1} * \eta' - \mu^{k+1}e\| < \mu^{k+1}r_1^2 = \mu^{k+1}(\alpha r_1)r, \tag{3.16}$$

and it follows from the first part of this theorem with R_1 and r_1 replaced by $R_2 = R/(\alpha r_1)$ and $(\alpha r_1)r$, respectively, that

$$\rho^{k+1} > R_2 > R_1. \qquad \square$$

Typical path-following algorithms (see [9] and [17]) take single Newton steps for a sequence of barrier problems $BP(\mu^1)$, $BP(\mu^2), \ldots$, where the barrier parameters $\{\mu^k\}$ are carefully selected so that the iteration points closely "follow" the central trajectory $x^1(\mu)$ defined at the beginning of this section. Although it is not readily apparent, the steps taken by Algorithm 1 are, in fact, Newton steps for the sequence of problems $\{PP(v^{k+1})\}$ under the projective transformations $\{T(x^k)\}$. Hence, our projective algorithm fits into the framework of a typical path-following algorithm. The only difference is that our algorithm uses the implicitly generated parameters $\{v^k\}$, instead of explicitly given $\{\mu^k\}$, to follow the path $x^2(v)$, which is identical to $x^1(\mu)$, and uses projective transformations. We note that in [5], Bayer and Lagarias

prove that under a suitable fixed projective transformation, which maps the level set corresponding to f^* to the "hyperplane at infinity," Karmarkar's original method becomes a global Newton method for minimizing a logarithmic barrier function.

To prove our claim about Algorithm 1 and Newton's method, consider problem $PP(v^{k+1})$ under the projective transformation $T(x^k)$; i.e.,

$$\begin{aligned} &\min \quad p(y, v^{k+1}) = n\ln(\hat{c}^T y - v^{k+1} y_n) - \sum_{j=1}^{n} \ln y_j - \ln(\det D) \\ (\tilde{P}(v^{k+1})) \quad &\text{s.t.} \quad \hat{A}y = 0, \\ &\qquad e^T y = n, \\ &\qquad y > 0, \qquad y \in R^n. \end{aligned}$$

Let $h = \hat{c} - v^{k+1} e_n$ and $B^T = [\hat{A}^T, e]$. Since the gradient and Hessian of the potential function $p(y, v^{k+1})$ at $y = e$ are, respectively,

$$g \equiv \nabla p(e, v^{k+1}) = (n/h^T y)h - e$$

and

$$G \equiv \nabla^2 p(e, v^{k+1}) = I - (n/(h^T y)^2) hh^T,$$

the Newton step for $\tilde{P}(v^{k+1})$ at the point e is

$$\Delta y^{Newton} = -\tilde{P} G^{-1} g,$$

where

$$\tilde{P} = I - G^{-1} B^T (B G^{-1} B^T)^{-1} B.$$

Using the Sherman–Morrison–Woodbury updating formula twice, we have that

$$G^{-1} = I + \alpha hh^T$$

and

$$(BG^{-1}B^T)^{-1} = (BB^T)^{-1} + \beta (BB^T)^{-1} Bhh^T B^T (BB^T)^{-1},$$

where α and β are scalars. Hence,

$$G^{-1} g = \gamma h - e,$$

and

$$\begin{aligned} \Delta y^{Newton} &= -\left[I - (I + \alpha hh^T)(\hat{P} + \beta \hat{P} hh^T \hat{P}) \right] (\gamma h - e) \\ &= -\gamma Ph + \sigma h + \tau \hat{P} h, \end{aligned} \tag{3.17}$$

where $\hat{P} = B^T (BB^T)^{-1} B$, $P = I - \hat{P}$, and γ, σ, and τ are scalars. Since $B\Delta y^{Newton} = 0$, $BP = 0$, and $B\hat{P} = B$, it follows from (3.17) that

$$\sigma Bh + \tau Bh = 0,$$

and hence that $\sigma = -\tau$. Therefore,

$$\Delta y^{Newton} = -(\gamma + \tau) Ph = -(\gamma + \tau)(c_p - v^{k+1} d_p),$$

which is the same as the step Δy computed by Algorithm 1, except for its length.

The following lemma provides the basis for our proof that Algorithm 1 terminates in at most $O(\sqrt{n}L)$ iterations.

LEMMA 3.1. *For* $k \geq 0$,

$$\frac{f^{k+1} - v^{k+1}}{f^k - v^{k+1}} = \frac{1 - r_1/R_1}{1 + \sigma r_1}$$

where

$$\sigma = -\frac{(c_p - v^{k+1}d_p)^T d_p}{\|c_p - v^{k+1}d_p\|} \quad \text{and} \quad -\frac{1}{R_1} < \sigma \leq 1 - \frac{1}{n}. \tag{3.18}$$

Besides, for $k \geq 1$,

$$\frac{f^k - v^{k+1}}{f^k - v^k} \leq \frac{r_1 + \sigma R_1}{1 + \sigma R_1}, \tag{3.19}$$

and

$$-\frac{1}{R_2} = -\frac{r_1}{R_1} < \sigma \leq 1 - \frac{1}{n}.$$

Proof. It directly follows from (3.14) and $y_n^{k+1} = 1 + e_n^T \Delta y = 1 + \sigma r_1$ that

$$\frac{f^{k+1} - v^{k+1}}{f^k - v^{k+1}} = \frac{1 - r_1^2/n}{y_n^{k+1}} = \frac{1 - r_1/R_1}{1 + \sigma r_1}.$$

Define

$$y(t) = e - t\frac{c_p - v^{k+1}d_p}{\|c_p - v^{k+1}d_p\|} \quad \text{and} \quad f(t) = \frac{\hat{c}^T y(t)}{y_n(t)}, \quad 0 < t < \infty.$$

Then, $y_n(t) = e_n^T y(t) = 1 + \sigma t$ and $f(R_1) = v^{k+1}$ by $E_{R_1}(v^{k+1})$. We previously showed (see (3.13)) that

$$\hat{c} - v^{k+1}e_n - \hat{A}^T \pi' = \frac{f^k - v^{k+1}}{n}(e - \Delta y) > 0,$$

where Δy is defined by (2.11). Since $y^T(t)\hat{A}^T = 0$ and $e^T \Delta y = 0$,

$$\begin{aligned}
(\hat{c} - v^{k+1}e_n)^T y(t) &= \frac{f^k - v^{k+1}}{n} y^T(t)(e - \Delta y) \\
&= \frac{f^k - v^{k+1}}{n}\left(e + \frac{t}{r_1}\Delta y\right)^T (e - \Delta y) \\
&= \frac{f^k - v^{k+1}}{n}\left(n - \frac{t}{r_1}\|\Delta y\|^2\right) \\
&= (f^k - v^{k+1})\left(1 - \frac{r_1 t}{n}\right) \\
&= (f^k - v^{k+1})\left(1 - \frac{t}{R_1}\right),
\end{aligned}$$

so

$$\frac{(f(t) - v^{k+1})y_n(t)}{f^k - v^{k+1}} = \frac{(\hat{c} - v^{k+1}e_n)^T y(t)}{f^k - v^{k+1}} = 1 - \frac{t}{R_1}. \tag{3.20}$$

For $0 < t < R_1$, the right-hand side of (3.20) is positive. Due to the continuity of $y_n(t)$, we know that $y_n(R_1) > 0$, and so $\sigma > -(1/R_1)$. Hence,

$$-\frac{1}{R_1} < \sigma \leq |\sigma| \leq \|d_p\| = \|Pe_n\| \leq \|P_e e_n\| = 1 - \frac{1}{n},$$

where the upper bound on $|\sigma|$ follows from the Cauchy–Schwarz inequality.

If $k \geq 1$, then by Theorem 1, we know that

$$\rho^k > R_2,$$

where $R_2 = R_1/r_1 = R/(\alpha^2 r)$. By Lemma 2.1 (i), we then have that

$$R_2 \leq \rho^k \leq \|y - e\| \quad \text{for any } y \in H(v^k). \tag{3.21}$$

Since $f(0) = f^k > v^k$ and Δy is a descent direction, we can conclude that $f(t)$ has no singular point in $(0, R_2)$ and $f(R_2) \geq v^k$, because otherwise, there must exist a t_1, $0 < t_1 < R_2$, such that $f(t_1) = v^k$, i.e., $y(t_1) \in H(v^k)$, which is a contradiction to (3.21).

Hence, by (3.20) with $t = R_2 = R_1/r_1$ and $y_n(R_2) = 1 + \sigma R_2$,

$$\begin{aligned}
\frac{f^k - v^{k+1}}{f^k - v^k} &\leq \frac{f^k - v^{k+1}}{f^k - f(R_2)} \\
&= \left(1 - \frac{f(R_2) - v^{k+1}}{f^k - v^{k+1}}\right)^{-1} \\
&= \left(1 - \frac{1 - 1/r_1}{y_n(R_2)}\right)^{-1} \\
&= \frac{r_1 + \sigma R_1}{1 + \sigma R_1}.
\end{aligned}$$

Since $\sigma > -(1/R_1)$, i.e., $1 + \sigma R_1 > 0$, from the inequality above, we know that $r_1 + \sigma R_1 > 0$. So

$$\sigma > -\frac{r_1}{R_1} = -\frac{1}{R_2}. \qquad \square$$

THEOREM 2. *Starting from an initial interior point x^0 that is close to the "central trajectory" and a lower bound v^0 such that $\rho^0 > R_1$, Algorithm 1 obtains an optimal solution in at most $O(\sqrt{n}L)$ iterations.*

Proof. Lemma 3.1 shows that for $k \geq 1$

$$\frac{f^{k+1} - v^{k+1}}{f^k - v^k} \leq \frac{1 - r_1/R_1}{1 + \sigma r_1} \frac{r_1 + \sigma R_1}{1 + \sigma R_1}.$$

Notice that both terms on the right-hand side of the above expression are less than one.

If $\sigma > (1 - r_1)/\sqrt{n}$, then

$$\begin{aligned}
\frac{f^{k+1} - v^{k+1}}{f^k - v^k} &< \frac{1 - r_1/R_1}{1 + \sigma r_1} < \frac{1}{1 + \sigma r_1} \\
&< \left(1 + \frac{(1 - r_1) r_1}{\sqrt{n}}\right)^{-1}.
\end{aligned}$$

Otherwise, $-(1/R_1) < \sigma < (1-r_1)/\sqrt{n}$, and we have

$$\begin{aligned}\frac{f^{k+1}-v^{k+1}}{f^k-v^k} &< \frac{r_1+\sigma R_1}{1+\sigma R_1} = \left(1+\frac{1-r_1}{r_1+\sigma R_1}\right)^{-1} \\ &< \left(1+\frac{1-r_1}{r_1+\sqrt{n}(1-r_1)/r_1}\right)^{-1} \\ &< \left(1+\frac{(1-r_1)r_1}{\sqrt{n}}\right)^{-1},\end{aligned}$$

where in the second inequality we have used $r_1R_1 = n$ and in the last inequality we have used $r_1^2 + (1-r_1)\sqrt{n} < \sqrt{n}$.

Since $(1-r_1)r_1 \geq r_1^2/2 \geq \alpha^2/2$ for $0 < r_1 \leq \frac{2}{3}$, it then follows that

$$\frac{f^{k+1}-v^{k+1}}{f^k-v^k} < \left(1+\frac{(1-r_1)r_1}{\sqrt{n}}\right)^{-1} < \left(1+\frac{\alpha^2}{2\sqrt{n}}\right)^{-1}.$$

Moreover, after the first iteration of Algorithm 1, $f^1 - v^1 < 2^{O(L)}$. Therefore, after $O(\sqrt{n}L)$ iterations,

$$f^k - v^k = (x^k)^T\eta^k < 2^{-L},$$

and the algorithm terminates after "rounding" x^k to an optimal vertex x^*. □

4. Initialization for Algorithm 1. Consider the standard form linear program:

$$(P_0)\qquad \begin{aligned} \min\quad & \tilde{c}^T x \\ \text{s.t.}\quad & \tilde{A}x = b, \\ & x \geq 0, \qquad x \in R^{n-2}. \end{aligned}$$

If initially we do not have a primal feasible interior point for problem (P_0), then we can introduce an artificial variable with a big-M cost to create an augmented problem with a feasible solution e. So, we can assume that there exists a primal feasible interior point x^0 for problem (P_0). Also, we assume that x^*, the optimal solution of (P_0), has the property that

$$\frac{1}{n-1}\sum_{j=1}^{n-2} x_j^* \leq M.$$

Although theoretically $M = 2^{O(L)}$ could be required, in practice much smaller values suffice. Let

$$b' \leftarrow \tilde{b}/M,$$

$$A_{(m+1)\times n} \leftarrow \begin{pmatrix} \tilde{A}D & 0 & -b' \\ e^T & 1 & -(n-1) \end{pmatrix} \quad \text{and} \quad c \leftarrow \begin{pmatrix} \tilde{c} \\ 0 \\ 0 \end{pmatrix},$$

where $D = \operatorname{diag}\{x_j^0/M\}$ and $e \in R^{n-2}$. We immediately obtain the following linear programming problem:

$$(P_1)\qquad \begin{aligned} \min\quad & c^T x \\ \text{s.t.}\quad & Ax = 0, \\ & x_n = 1, \\ & x \geq 0, \qquad x \in R^n, \end{aligned}$$

where $e \in R^n$ is a feasible interior point. Moreover, we have

$$Pe_n = 0,$$

since $Pe_n = P_A P_e e_n$ and

$$P_e e_n = -\frac{1}{n}(e, 1, -(n-1))^T,$$

which is parallel to the last row of A. Therefore, in order to satisfy the initial condition,

$$\rho^0 = \frac{f^0 - v^0}{\|P\hat{c} - v^0 P e_n\|} = \frac{f^0 - v^0}{\|P\hat{c}\|} > R_1,$$

we just take v^0 small enough. Then, we can begin our Algorithm 1 from the primal feasible interior point e and the lower bound v^0.

There is another way to initiate Algorithm 1. Starting from an arbitrary interior feasible solution we can obtain a feasible solution close to the central path by taking "centering steps." In general, this can require as many as $O(nL)$ iterations. However, if the primal-dual potential function

$$p(x, \eta) = (n + \nu)\ln(x^T\eta) - \sum_{j=1}^{n} \ln x_j - \sum_{j=1}^{n} \ln \eta_j \tag{4.1}$$

with $\nu = 1$ initially satisfies $p(x^0, \eta^0) = O(L)$, then only $O(L)$ iterations are required. (See [23] for details.) This gives rise to a two-phase algorithm that finds a point close to the "central path" in the first phase and reduces to Algorithm 1 in the second phase, and that is closely related to a special case of the $O(\sqrt{n}L)$-step generic algorithm described in [8].

Because this generic algorithm is based solely upon reducing the potential function (4.1) with $\nu = \sqrt{n}$, long steps can be taken by performing a linesearch. These steps fluctuate between path-following and centering steps, while the steps taken by the two-phase method are all at first centering steps and then all short path-following ones. Another long-step modification of Algorithm 1 has been proposed by Anstreicher [3].

5. The anticipated behavior of the algorithm. In this section, we show that under a certain probabilistic assumption, the expected number of iterations required by Algorithm 1 is $O(L)$. In practice, projective interior point methods work quite well, and the number of iterations required by them tends to be nearly, but not completely, independent of n (see Adler, Karmarkar, Resende, and Veiga [1], and Karmarkar and Ramakrishnan [14]). In order to formally explain this phenomenon, several researchers have analyzed the so-called "anticipated behavior" of interior point methods. (See, for example, Gonzaga and Todd [11] and Ye [25]). In these analyses, the probabilistic assumption is made that certain directions that arise in the algorithm under study are uniformly distributed on the unit sphere. These analyses obtain that the average number of iterations required by each of the interior point methods studied is much better than the worst-case bound for the method, but still dependent on n.

Under a similar but more reasonable assumption, which rules out directions that cannot occur, we now prove that the expected number of iterations required by our algorithm is $O(L)$. However, various interior point methods that have been implemented appear to take a number of iterations, which grows slightly as the number

of variables of the linear programming problem being solved is increased. Hence, our result raises the question of whether the probabilistic assumptions that underlie all of the anticipated behavior analyses, including our own, are valid, or the average number of iterations taken by the algorithm that we proposed could really be independent of n in practice. Further research is needed to answer this question. In the following, we prove the result mentioned above, which is based on the assumption that the normalized step directions of the algorithm are uniformly distributed over that portion of the unit sphere that is allowable. Unfortunately, our analysis (like the analyses in [11] and [25]) does not obtain a rigorous expected value bound for the algorithm, because the assumption that search directions are uniformly distributed cannot be justified after the first step of the algorithm.

In preparation for our probability analysis, recall that the "σ" defined in (3.18) is a real number satisfying $-(1/R_1) < \sigma \leq 1 - 1/n$. It is easy to verify that the upper bound is tight. Obviously, the value of σ is different from iteration to iteration, so we denote it as σ^k at the kth iteration. From our analysis, when $\sigma^k = O(1/\sqrt{n})$ the algorithm encounters the worst case. But it is unlikely that σ^k is $O(1/\sqrt{n})$ on all iterations and for all problems. It is more reasonable to assume that σ^k is randomly distributed in $(-(1/R_1), 1 - \frac{1}{n}]$. If that is the case, we have the following result.

LEMMA 5.1. *Assume that $\{\sigma^k\}$ is a series of independent random variables with the same distribution on the interval $(-(1/R_1), 1 - \frac{1}{n}]$, and either*

$$P(\sigma^k < 0) \geq p > 0, \tag{5.1}$$

or

$$E(\sigma^k | \sigma^k > 0) \geq \mu > 0, \tag{5.2}$$

where μ and p are constants independent of n and the input data.

Then, the expected number of iterations required by Algorithm 1 is $O(L)$.

Proof. Define $\mathcal{K} = \min\{K | \sum_{k=1}^{K} z^k < -2L\}$, where

$$z^k \equiv \log \frac{1 - r_1/R_1}{1 + \sigma^k r_1} \frac{r_1 + \sigma^k R_1}{1 + \sigma^k R_1} < 0. \tag{5.3}$$

It immediately follows from Lemma 3.1 that

$$\frac{f^{k+1} - v^{k+1}}{f^k - v^k} \leq 2^{z^k},$$

for $k \geq 1$. Then, after $\mathcal{K}$ iterations, the algorithm terminates with

$$f^{\mathcal{K}+1} - v^{\mathcal{K}+1} < 2^{-L}.$$

Since we assume that $\{\sigma^k\}$ is a sequence of independent random variables, so is $\{z^k\}$, and $\mathcal{K}$ is a stopping time. From our deterministic analysis $E\mathcal{K} = O(\sqrt{n}L) < \infty$; hence, due to Wald's equation,

$$E\left\{\sum_{k=1}^{\mathcal{K}} z^k\right\} = E\mathcal{K}Ez^1.$$

By the definition of $\mathcal{K}$,

$$\sum_{k=1}^{\mathcal{K}-1} z^k \geq -2L.$$

Hence, $E\mathcal{K}Ez^1 - Ez^1 \geq -2L$. Therefore, if $Ez^1 < 0$,

$$E\mathcal{K} \leq \frac{2L}{-Ez^1} + 1. \tag{5.4}$$

Set α so that $r_1 = \frac{2}{3}$. Then, it follows from (5.3) that, for $\sigma^k \leq 0$,

$$z^k < \log \frac{r_1 + \sigma^k R_1}{1 + \sigma^k R_1} \leq \log r_1 = \log \frac{2}{3}. \tag{5.5}$$

Hence,

$$E(z^k | \sigma^k \leq 0) \leq \log \frac{2}{3} < 0,$$

and as $z^k \leq 0$,

$$\begin{aligned} E\{z^k\} &= P(\sigma^k > 0)E\{z^k|\sigma^k > 0\} + P(\sigma^k \leq 0)E\{z^k|\sigma^k \leq 0\} \\ &\leq P(\sigma^k \leq 0)E\{z^k|\sigma^k \leq 0\} \leq p \log \frac{2}{3} < 0 \end{aligned}$$

if (5.1) holds. Therefore, from (5.4) we have $E(\mathcal{K}) = O(L)$. If (5.2) holds, since $0 < \sigma^k < 1$,

$$\begin{aligned} z^k &\leq \log \frac{1}{1 + \sigma^k r_1} \leq -\sigma^k r_1 + \frac{1}{2}(\sigma^k r_1)^2 \\ &\leq -\left(r_1 - \frac{1}{2}r_1^2\right)\sigma^k = -\frac{4}{9}\sigma^k. \end{aligned} \tag{5.6}$$

Hence, from (5.5) and (5.6),

$$\begin{aligned} E\{z^k\} &= P(0 < \sigma^k < 1)E\{z^k|\sigma^k > 0\} + P(\sigma^k \leq 0)E\{z^k|\sigma^k \leq 0\} \\ &\leq \max\left(-\frac{4}{9}\mu, \log \frac{2}{3}\right) = -\min\left(\frac{4}{9}\mu, \log \frac{3}{2}\right) < 0. \end{aligned}$$

Again, from (5.4) we have $E\mathcal{K} = O(L)$. □

Now, we formally describe our probabilistic assumption. Let

$$C_l = \left\{ g \mid g = \sum_{j \neq l} \lambda_j (ne_j - e), \lambda_j \geq 0, 1 \leq j \leq n, j \neq l \right\}.$$

Then, the simplex $\{ x | \ e^T x = n, x \geq 0 \} \subseteq \{e + g | \ g \in \cup_{l=1}^n C_l \} = \{ x \mid e^T x = n \}$. The linear program (P) is bounded if and only if the fractional linear program $(\hat{P}_k)$ is bounded, and if and only if there is no feasible descent ray g of $(\hat{P}_k)$ such that $g \in C_n$. Since our step directions are descent directions, and we assume that (P) is bounded, no step direction in our algorithm belongs to C_n.

Let S be the unit sphere in $\{ x \mid e^T x = 0 \}$ and $H_l = S \cap C_l$, for $l = 1, 2, \ldots, n$. Notice that S is an $(n-2)$-dimensional manifold and $S = \cup_{l=1}^n H_l$, since $S \subseteq \cup_{l=1}^n C_l = \{x \mid e^T x = 0\}$.

Now let us assume that the step directions of Algorithm 1, g^k, are uniformly distributed on H', where $H' = S - H_n = \cup_{l=1}^{n-1} H_l$. From the definition of σ^k in (3.18) and $d_p = Pe_n$, we know that

$$\sigma^k = g_n^k,$$

where g^k is the step direction of Algorithm 1, i.e.,

$$g^k = -\frac{c_p - v^{k+1} d_p}{\|c_p - v^{k+1} d_p\|}.$$

Define the hemisphere $H^- = \{\, g \mid g \in S, g_n \leq 0\}$. Obviously, $H_n \subset H^-$. Since g^k is uniformly distributed on H',

$$P(g_n^k \leq 0) = \frac{|H^- \cap H'|}{|H'|},$$

where $|\cdot|$ denotes the area of a manifold. Since

$$|H_1| = \cdots = |H_n| \quad \text{and} \quad |S| = \sum_{l=1}^{n} |H_l|,$$

$$|H'| = |\cup_{l=1}^{n-1} H_l| = (n-1)|H_n|,$$

and

$$|H^- \cap H'| = |H^- - H_n| = \frac{1}{2}|S| - |H_n| = \left(\frac{n}{2} - 1\right)|H_n|.$$

Consequently,

$$P(\sigma_k \leq 0) = P(g_n^k \leq 0) = \frac{n-2}{2(n-1)} \geq \frac{1}{4}, \tag{5.7}$$

for $n \geq 3$.

Now, from Lemma 5.1 and (5.7), it follows that our algorithm terminates in $O(L)$ iterations on average!

THEOREM 3. *Assume that the sequence of step directions of Algorithm* 1, $\{g^k\}$, *is a sequence of independent random vectors that are uniformly distributed on* H'. *Then, the expected number of iterations required by Algorithm* 1 *is* $O(L)$.

Appendix. In this appendix, we show that even though the potential function is nonconvex, it shares a basic property with the logarithmic barrier function. Consider the following problem:

$$(P_v) \qquad \begin{aligned} \min \quad & p(x,v) = n \ln(c^T x - v) - \sum_{j=1}^{n} \ln x_j \\ \text{s.t.} \quad & Ax = b, \\ & x > 0, \qquad x \in R^n, \end{aligned}$$

where $v < f^* = \min\{c^T x \mid \ Ax = b, x \geq 0\}$ and $b \neq 0$.

THEOREM A. *A feasible solution* x *of* (P_v) *is an optimal solution if and only if* x *satisfies the first-order optimality conditions for* (P_v). *Besides, the optimal solution of* (P_v) *is unique.*

Proof. Necessity is a direct result of the Karush–Kuhn–Tucker conditions. To prove sufficiency, we show that if x satisfies the first-order optimality condition

$$\nabla p(x) = A^T\lambda,$$

then for any feasible direction $y \neq 0$ (i.e., $Ay = 0$),

$$y^T H(x) y > 0,$$

where

$$\nabla p(x) = \frac{n}{(cx - v)} c - D^{-1}e,$$

and

$$H(x) = D^{-2} - \frac{n}{(c^T x - v)^2} cc^T,$$

are, respectively, the gradient and Hessian of the potential function.

From the first-order conditions, we have $\nabla^T p(x) y = \lambda Ay = 0$, i.e.,

$$\frac{n}{c^T x - v} c^T y - e^T D^{-1} y = 0.$$

Let $z = D^{-1}y$, then

$$y^T H(x)^T y = y^T D^{-2} y - \frac{n}{(cx - v)^2}(c^T y)^2 = z^T z - \frac{1}{n}(e^T z)^2 \geq 0,$$

where the last inequality becomes an equality if and only if $z = ke$ (k is a scalar and $k \neq 0$). But equality is impossible, since the right-hand side $b \neq 0$ implies that $y \neq kx$ for any scalar $k \neq 0$. Therefore,

$$y^T H(x) y > 0. \qquad \square$$

We note that for (P_v) to have a solution, the set of optimal solutions to the linear program with which (P_v) is associated must be bounded, as is the case for the linear program (P) under our assumptions on (P) and (D). Under the assumption that the feasible region of (P_v) is bounded, Theorem A can also be proved using the approach in [12].

Acknowledgment. We wish to thank Kurt Anstreicher for pointing out that Theorem 2 could be proved without recourse to a potential function, and two anonymous referees for several useful comments.

REFERENCES

[1] I. Adler, N. Karmarkar, M. G. C. Resende, and G. Veiga, *An implementation of Karmarkar's algorithm for linear programming*, Math. Programming, 44 (1989) pp. 297–335.

[2] K. M. Anstreicher, *A monotonic projective algorithm for fractional linear programming*, Algorithmica, 1 (1986), pp. 483–498.

[3] ———, *Strict Monotonicity and Improved Complexity in the Standard Form Projective Algorithm for Linear Programming*, manuscript, C.O.R.E., Université Catholique de Louvain, Louvain-La-Neuve, Belgium, 1990; Math. Programming, to appear.

[4] D. A. BAYER AND J. C. LAGARIÀS, *The nonlinear geometry of linear programming*, II *Legendre transform coordinates and central trajectories*, Trans. Amer. Math. Soc., 314 (1989), pp. 527–581.

[5] ———, *Karmarkar's linear programming algorithm and Newton's method*, Math. Programming, 50 (1991), pp. 291–330.

[6] G. DE GHELLINCK AND J. P. VIAL, *A polynomial Newton method for linear programming*, Algorithmica, 1 (1986), pp. 425–453.

[7] D. GOLDFARB AND D. SHAW(XIAO), *A primal projective interior point method for linear programming*, Math. Programming, 51 (1991), pp. 17–41.

[8] ———, *On the complexity of a class of projective interior point methods*, manuscript, Department of Industrial Engineering and Operations Research, Columbia University, New York, NY, 1991, Math. Oper. Res., to appear.

[9] C. GONZAGA, *An algorithm for solving linear programming in $O(n^3L)$ operations*, in Progress in Mathematical Programming: Interior Point and Related Methods, N. Megiddo ed., Springer-Verlag, New York, 1988, pp. 1–28.

[10] ———, *Large-steps path-following methods for linear programming, Part* II: *Potential reduction method*, SIAM J. Optimization, 1 (1991), pp. 280–292.

[11] C. C. GONZAGA AND M. J. TODD, *An $O(\sqrt{n}L)$-iteration large-step primal-dual affine algorithm for linear programming*, SIAM J. Optimization, 2 (1992), pp. 349–359.

[12] H. IMAI, *On the convexity of the multiplicative version of Karmarkar's potential function*, Math. Programming, 40 (1988), pp. 29–32.

[13] N. KARMARKAR, *A new polynomial time algorithm for linear programming*, Combinatorica, 4 (1984), pp. 373–395.

[14] N. KARMARKAR, AND K. G. RAMAKARISHNAN, *Implementation and computation results of Karmarkar's algorithm for linear programming, using an iterative method for computing projections*, extended abstract, AT&T Bell Laboratories, Murray Hill, NJ, 1988.

[15] M. KOJIMA, S. MIZUNO, AND A. YOSHISE, *A polynomial-time algorithm for a class of linear complementarity problems*, Math. Programming, 44 (1989), pp. 1–26.

[16] N. MEGIDDO, *Pathways to the optimal set in linear programming*, in Progress in Mathematical Programming: Interior Point and Related Methods, N. Megiddo ed., Springer-Verlag, New York, 1988, pp. 131–158.

[17] R. C. MONTEIRO AND I. ADLER, *An $O(n^3L)$ interior point algorithm. Part* I: *Linear programming*, Math. Programming, 44 (1987), pp. 27–42.

[18] M. PADBERG, *A different convergence proof of the projective method for linear programming*, Oper. Res. Lett., 4 (1986), pp. 373–395.

[19] J. RENEGAR, *A polynomial-time algorithm, based on Newton's method, for linear programming*, Math. Programming, 40 (1988), pp. 59–93.

[20] C. ROOS, *New trajectory-following polynomial-time algorithm for linear programming*, J. Optimization Theory Appl., 63 (1989), pp. 433–458.

[21] C. ROOS AND J. P. VIAL, *A polynomial method of approximate centers for linear programming*, Math. Programming, 54 (1992), pp. 295–305.

[22] ———, *Long steps with the logarithmic barrier function in linear programming*, in Economic Decision Making: Games, Econometrics and Optimization, J. Gabszewicz, J.-F. Richard, and L. Wolesey, eds., Elsevier Science Publishers B.V., Amsterdam, 1990, pp. 433–441.

[23] D. SHAW(XIAO), *Projective Interior Points Methods for Linear Programming*, Ph.D. Thesis, Columbia University, New York, New York, 1991.

[24] M. J. TODD AND B. P. BURRELL, *An extension of Karmarkar's algorithm for linear programming using dual variables*, Algorithmica, 1 (1986), pp. 409–424.

[25] Y. YE, *Comparative analysis of affince scaling algorithms based on simplifying assumptions*, Math. Programming, 52 (1991), pp. 405–414.

SIAM J. OPTIMIZATION
Vol. 4, No. 1, pp. 86–107, February 1994
©1994 Society for Industrial and Applied Mathematics
004

FAST APPROXIMATION SCHEMES FOR CONVEX PROGRAMS WITH MANY BLOCKS AND COUPLING CONSTRAINTS*

MICHAEL D. GRIGORIADIS† AND LEONID G. KHACHIYAN†

Abstract. This paper presents block-coordinate descent algorithms for the approximate solution of large structured convex programming problems. The constraints of such problems consist of K disjoint convex compact sets B^k called blocks, and M nonnegative-valued convex block-separable inequalities called coupling or resource constraints. The algorithms are based on an exponential potential function reduction technique. It is shown that feasibility as well as min-max resource-sharing problems for such constraints can be solved to a relative accuracy ε in $O(K \ln M(\varepsilon^{-2} + \ln K))$ iterations, each of which solves K block problems to a comparable accuracy, either sequentially or in parallel. The same bound holds for the expected number of iterations of a randomized variant of the algorithm which uniformly selects a random block to process at each iteration. An extension to objective and constraint functions of arbitrary sign is also presented. The above results yield fast approximation schemes for a number of applications such as problems with additively separable functions, generalized concurrent flows with side constraints, linear and nonlinear supply-sharing transportation networks, and deterministic equivalents of certain two-stage stochastic programs. Another consequence of this analysis is that, for a fixed relative accuracy, the approximate solution of matrix games is in *NC*.

Key words. approximation algorithm, block angular optimization, complexity, exponential potential function, fully polynomial time approximation scheme, linear programming, multicommodity flow, nonlinear programming, parallel computation, randomized algorithm, structured optimization

AMS subject classifications. 68Q25, 90C05, 90C30

1. Introduction.

Large mathematical programming models which arise in practice frequently exhibit some structure in their constraints. The most common of these is the *block angular* structure. It consists of K sets of "block" constraints and a set of M "coupling" constraints, so that if the coupling constraints are sufficiently relaxed, the problem decomposes into K simpler independent problems. Among the earliest methods specifically designed to take advantage of this structure for linear problems are the Dantzig–Wolfe decomposition principle [2] and Rosen's partitioning method [19]. Currently there is a renewed interest in complexity results for solving large structured optimization problems, mainly prompted by advances in parallel computation and potential-function reduction methods for linear and nonlinear programming. In this paper we present and analyze the complexity of a block-coordinate potential-descent method for a class of such problems.

*Received by the editors July 17, 1992; accepted for publication November 13, 1992. The preparation of this paper was supported in part by the National Science Foundation under grant CCR-9208539. The present paper first appeared as a DIMACS Technical Report [8] when the second author was supported in part by the Center for Discrete Mathematics and Theoretical Computer Science (DIMACS), a National Science Foundation Science and Technology Center, Rutgers University, New Brunswick, New Jersey.

†Department of Computer Science, Hill Center, Rutgers University, New Brunswick, New Jersey 08903.

1.1. The problem. We consider the following block-angular convex program:

$$\lambda^* = \min\left\{ \lambda \;\middle|\; \begin{array}{l} \sum_{k=1}^{K} f_m^k(x^k) \le \lambda, \quad m = 1,\dots,M \\ x^k \in B^k, \quad k = 1,\dots,K \end{array} \right\},$$

where B^k are nonempty convex compact sets, and $f_m^k : B^k \to \mathbb{R}$ are nonnegative continuous convex functions. The sets B^k are called *blocks*, the variables $x^k \in B^k$ are called *block coordinates*, and the M inequalities are called *coupling* or *resource constraints*. We shall use the following notation: $x = (x^1,\dots,x^K)$, $B = B^1 \times \cdots \times B^K$, $f^k = (f_1^k,\dots,f_M^k) : B^k \to \mathbb{R}^M$, $f = \sum_{k=1}^K f^k : B \to \mathbb{R}^M$, and $e \in \mathbb{R}^M$ the vector of all ones. With this notation, the above problem can be written as

$$\lambda^* = \lambda(x^*) = \min_{x \in B}\{\, \lambda(x) = \max\{\, f_1(x),\dots,f_M(x)\,\}\,\}, \tag{1.1}$$

or simply written as follows:

$$\lambda^* = \min\{\, \lambda \mid f(x) \le \lambda e,\ x \in B\,\}. \tag{$\mathcal{P}$}$$

We shall assume throughout that $M \ge 2$, and that $\lambda_k^* = \min\{\, \lambda \mid f^k(x^k) \le \lambda e,\ x^k \in B^k\,\}$ is positive for all $k = 1,\dots,K$; that is, there are no blocks B^k that can be discarded from $(\mathcal{P})$ without affecting the value λ^*. Our main focus shall be to state an algorithm which, for a given relative accuracy $\varepsilon \in (0,1]$, computes an *ε-approximate solution* of $(\mathcal{P})$:

$$f(x) \le (1+\varepsilon)\lambda^* e, \qquad x \in B. \tag{$\mathcal{P}_\varepsilon$}$$

Combined with bisection search, such an algorithm can be used to approximately solve the general block-angular optimization problem

$$f^* = \min\{\, f_0(x) \mid f(x) \le e,\ \ x \in B\,\}, \tag{1.2}$$

with a block-separable nonnegative convex objective function $f_0 : B \to \mathbb{R}_+$. This entails only a logarithmic increase in the number of iterations. The nonnegativity assumption on the objective and constraint functions can be relaxed, albeit at some cost. In many problems the functions are additively separable, which leads to algorithmic simplifications and efficiencies. We consider these cases as applications in §9.2.

1.2. Block-coordinate descent. Suppose that a point $x_0 \in B$ is known and let $\mu \ge \lambda(x_0)$ be a fixed scalar. Then, by the nonnegativity of the functions f^k, any solution x to $(\mathcal{P})$ also belongs to $B_\mu = B_\mu^1 \times \cdots \times B_\mu^K$, where

$$B_\mu^k = \{\, x^k \in B^k \mid f^k(x^k) \le \mu e\,\}, \qquad k = 1,\dots,K. \tag{1.3}$$

From von Neumann's saddlepoint theorem (see, e.g., Rockafellar [22, p. 393]),

$$\lambda^* = \min_{x \in B_\mu} \max_{p \in P} L(x,p) = \max_{p \in P} \min_{x \in B_\mu} L(x,p),$$

where $P = \{p \in \mathbb{R}^M \mid p \geq 0, \quad e^T p = 1\}$ is the unit simplex, and $L(x,p) = \sum_{k=1}^K L^k(x^k,p) = \sum_{k=1}^K p^T f^k(x^k) = p^T f(x)$ is the Lagrangian of $(\mathcal{P})$. An immediate consequence of this theorem is the well-known duality relation:

$$(\mathcal{D}) \qquad \lambda^* = \max_{p \in P} \left\{ g_\mu(p) = \sum_{k=1}^K g_\mu^k(p) \;\middle|\; p \in P \right\},$$

where

$$(\mathcal{P}^k) \qquad g_\mu^k(p) = \min\{ L^k(x^k,p) = p^T f^k(x^k) \mid x^k \in B_\mu^k \}.$$

Thus, for fixed μ and *price vector* $p \in P$, the computation of $g_\mu(p)$ decomposes into K independent *block problems* $(\mathcal{P}^k)$. This observation leads to a variety of decomposition methods which maintain $x \in B$, $p \in P$, the scalar μ, and a subset of blocks $\mathcal{K} \subseteq \{1,\ldots,K\}$ selected for recomputation, and perform at each iteration the following two basic steps:

Step 1. (Block optimization.) For each $k \in \mathcal{K}$, use a "block solver" to compute an approximate solution $\hat{x}^k \in B_\mu^k$ to the block problem $\mathcal{P}^k$.

Step 2. (Coordination.) For each $k \in \mathcal{K}$ replace the current x^k by a convex combination of x^k and the new point $\hat{x}^k$. Use an appropriate method to update p, μ, and $\mathcal{K}$.

The terms "iteration" and "coordination step" are used interchangeably. Typically, the amount of effort expended on the block problems dominates the cost of coordination. The block problems can be solved in parallel.

This general scheme includes a number of methods based on different ideas for updating p. These have been studied mainly for the case $\mu = +\infty$. Among the earliest contributions are the Brown–Robinson method for solving polyhedral games [20], the scheme of Everett [3], and two-level price decomposition methods (see, e.g., Lasdon [15]). In the past, straightforward implementations of these methods were found to be practical for computing reasonable estimates of solutions, but inefficient for attaining high accuracy, especially for linear models. A lower bound on the convergence rate for the latter case was given by Khachiyan [13]. Also included in this general scheme are methods that use a potential function $\Phi : f(x) \to \mathbb{R}$, smooth with respect to $f(x)$, as an approximation to a given objective function. Such methods determine the vector $p = p(f(x))$ by linearizing Φ with respect to $f(x)$ at the current iterate and select an appropriate convex combination in Step 2 so as to approximately minimize the potential within the blocks. In this respect they are reminiscent of successive (under)relaxation and Frank–Wolfe [4] methods. An early example is the method of Fratta, Gerla, and Kleinrock [5] which minimizes a rational potential function for the multicommodity flow problem in the context of routing in communication networks.

Recently, Schultz and Meyer [24] studied the convergence of block-coordinate descent based on a logarithmic potential function. Their coordination step includes a K-dimensional search to minimize this potential over all convex combinations of block coordinates in Step 2. By further exploiting the coarse parallelism inherent in Step 1, they were able to solve several large linear multicommodity network flow problems with $K = 11$ to an acceptable accuracy. Shahrokhi and Matula [21] were the first to use an exponential potential function for the concurrent flow problem[1] with

[1] This problem is to minimize the maximum relative arc overload in a capacitated network to accommodate a K-commodity flow with prescribed supplies and demands for each commodity.

uniform capacities, and to prove that the resulting method is a fully polynomial-time approximation scheme. Their bounds were subsequently improved by Tardos [25] and by Klein, Stein, and Tardos [14]. Leighton et al. [16] further extended [21], improved the bounds for nonuniform integer capacities, and suggested a randomized algorithm for selecting blocks (commodities) at each iteration.

1.3. Results. In this paper we present a block-coordinate descent method for solving problem $(\mathcal{P}_\varepsilon)$ for $\varepsilon \in (0, 1]$ in

$$O(K(\ln M)(\varepsilon^{-2} + \ln K)) \tag{1.4}$$

coordination steps. Each step generates K parallel calls to block solvers which compute solutions to problems $(\mathcal{P}^k)$ for some μ, one for each block, to a relative accuracy comparable to ε. We also state a uniform randomized variant of this method which runs in an expected number (1.4) coordination steps, each of which generates, on the average, two block-solver calls for randomly selected blocks. This randomized variant for problem $(\mathcal{P})$ is simpler than that suggested in [16] for concurrent flows, and it improves the expected number of iterations by a factor of ε^{-1}. Our general bounds are also lower than those independently obtained by Goldberg [7], who uses a recent result of Karp [12] and a similar randomization scheme for concurrent flows, as well as those of Plotkin, Shmoys, and Tardos [18] for packing and covering linear programs.

The bounds (1.4) on the number of block optimizations required to obtain an approximate solution $x^1, \ldots, x^K$ of $(\mathcal{P})$ hold for arbitrary nonnegative convex functions $f^k(x^k)$ and compact sets B^k, even if the x^k are of infinite dimension. These are the only known bounds for such general problems. Moreover, in many applications it is possible to simplify the sets B^k_μ, by sufficiently relaxing the constraints $f(x) \leq \mu e$. We shall show that relaxing these constraints by a fixed factor can only increase the bound on the number of coordination steps by essentially the same factor. This offers opportunities for trade-offs, such as replacing $f^k(x^k)$ by box constraints for additively-separable functions, which ultimately reduce the overall running time for various applications.

We have organized the paper as follows. Section 2 defines the particular exponential potential function Φ and price vector p to be used throughout. The function Φ, which generalizes the potentials of [21] and [16] for concurrent flows, can be interpreted as the "free energy" of a system with a fixed "temperature" $t > 0$ and "energy levels" $-f_m(x)$, $m = 1, \ldots, M$; the vector p then corresponds to the Gibbs distribution for such a system.

Section 3 describes an approximate block solver for implementing our coordinate descent algorithms. Section 4 presents Algorithm $\mathcal{A}$ which solves problem $(\mathcal{P})$ to a given *absolute* accuracy σ, starting from a normalized approximation x_0. This algorithm, which is invoked by all others, is analyzed in §5. Section 6 describes a randomized variant, Algorithm $\mathcal{R}$, and proves a bound on the expected number of iterations.

The next algorithm, described in §7, is a ternary search procedure, which gradually decreases t and calls either one of the previous two algorithms to solve $(\mathcal{P})$ to a given *relative* accuracy ε, starting from an arbitrary $x_0 \in B$. The resulting iteration bound depends logarithmically on the relative accuracy of x_0, a satisfactory level of performance for many practical instances. Section 8 presents a greedy initialization procedure to construct a sufficiently accurate initial approximation that ensures the claimed fully-polynomial bound (1.4) on the number of iterations.

The last section, 9, discusses some computational issues and applications, such as specializations to additively separable functions, concurrent flows, (gain-loss) multicommodity flows with side constraints across several arcs, uniformly sparse linear programs with generalized upper bounds, linear, quadratic, and min-max supply sharing generalized transportation networks. Also discussed in this section are optimization problems with functions of arbitrary sign, deterministic equivalents of large-scale two-stage stochastic programs with discrete finite distribution of input parameters, and a fast parallel approximation algorithm for matrix games, implying that for a fixed accuracy the latter are in *NC*.

2. Potential function and price vector. In stating our algorithms and their analysis, we shall work with a (convex) potential function of the form:

$$\Phi_t(x) = \ln \sum_{m=1}^{M} \exp \frac{f_m(x)}{t}, \qquad x \in B, \tag{2.1}$$

where $t > 0$ is a small positive fixed parameter to be specified later in this section. A useful property of (2.1) is that the difference of the potential at any two points $x, y \in B$ can be written as:

$$\Phi_t(y) - \Phi_t(x) = \ln \sum_{m=1}^{M} p_m \exp\left\{ \frac{f_m(y) - f_m(x)}{t} \right\}$$

where

$$p_m = p_m(x,t) = \frac{\exp\{ f_m(x)/t \}}{\sum_{i=1}^{M} \exp\{ f_i(x)/t \}}, \qquad m = 1, \ldots, M. \tag{2.2}$$

Our algorithm for $(\mathcal{P})$ will use this $p \in P$ as the price vector within the context of the block-coordinate descent strategy outlined in §1.2.

Another property of $\Phi_t(x)$ is the following:

$$\frac{\lambda(x)}{t} \leq \Phi_t(x) \leq \frac{\lambda(x)}{t} + \ln M \quad \text{for all } x \in B, \tag{2.3}$$

which is a direct consequence of the definition of $\lambda(x)$ in (1.1). Therefore,

$$\frac{\lambda^*}{t} \leq \Phi_t^* \leq \frac{\lambda^*}{t} + \ln M, \tag{2.4}$$

where $\Phi_t^* = \min_{x \in B} \Phi_t(x)$ and λ^* is the unknown optimal value of $(\mathcal{P})$. These two inequalities immediately imply the following lemma.

LEMMA 1. *Let $t = \sigma/(\delta + \ln M)$, where $\sigma > 0$ is a given absolute tolerance and δ is an arbitrary positive scalar. If $x \in B$ minimizes $\Phi_t(x)$ over B to an absolute accuracy δ, i.e., $\Phi_t(x) \leq \Phi_t^* + \delta$, then x minimizes $\lambda(x)$ over B to an absolute accuracy σ, i.e., $\lambda(x) \leq \lambda^* + \sigma$.*

We can thus replace the problem of approximately minimizing $\lambda(x)$ over $x \in B$ by the problem of approximately minimizing $\Phi_t(x)$ over $x \in B$. We shall use this observation in §4 to state a simple block-coordinate descent algorithm for solving $(\mathcal{P})$ to a given absolute accuracy. This algorithm requires a block solver which we define next.

3. Approximate block solvers. Due to the generality of the block problems $(\mathcal{P}^k)$:

$$\min\{p^T f^k(x^k) \mid x^k \in B^k_\mu\},$$

stated for fixed $\mu > 0$ and $p \in P$, it is unrealistic to expect that they can be solved exactly. It is thus necessary to ascertain the quality of a solution $\hat{x}^k$ computed for $(\mathcal{P}^k)$. We shall address this need by defining an approximate block solver as follows.

Let $\epsilon' > 0$ and $q > 1$ be given finite relative *block-optimization* and *block-feasibility* tolerances, respectively. We define an (ε', q)-*good block solver* for $(\mathcal{P}^k)$ to be a finite-time procedure $\mathcal{BS}(f^k, p, \mu)$ that attempts to find a solution $\hat{x}^k$ such that

$$\hat{x}^k \in B^k_{\mu q} = \{\, x^k \in B^k \mid f^k(x^k) \le \mu q e \,\} \tag{3.1}$$

and

$$p^T f^k(\hat{x}^k) \le (1+\varepsilon') g^k_\mu(p) = (1+\varepsilon') \min\{\, p^T f^k(x^k) \mid x^k \in B^k_\mu \,\}. \tag{3.2}$$

We assume further that $\mathcal{BS}$ has the following properties: It succeeds whenever $B^k_\mu \neq \emptyset$, it correctly reports all instances of $B^k_{\mu q} = \emptyset$, but behaves unreliably whenever $B^k_\mu = \emptyset$ and $B^k_{\mu q} \neq \emptyset$, returning any $\hat{x}^k$ or a fallacious report that $B^k_{\mu q} = \emptyset$.

Observe that since $B^k_\mu \subseteq B^k_{\mu q}$ for $q > 1$, it follows from (3.2) that for a fixed ε' the task of the block solver is simplified with increasing values of q.

The algorithms of §§4–7 will invoke procedure $\mathcal{BS}$ only for the case $B^k_\mu \neq \emptyset$. We shall also need to call $\mathcal{BS}$ for functions f^k scaled by some positive scalar ν. Since $\mathcal{BS}(\nu f^k, p, \mu) = \mathcal{BS}(f^k, p, \mu/\nu)$, the effect of such scaling on the parameters ϵ' and q can be ignored.

4. Solving problem $(\mathcal{P})$ to a given absolute accuracy. The algorithm, *Algorithm* $\mathcal{A}$, which we state in this section is an intrinsic component of the overall algorithms we shall develop in subsequent sections for solving $(\mathcal{P})$ to a given relative accuracy $\varepsilon > 0$ (problem $(\mathcal{P}_\varepsilon)$). Algorithm $\mathcal{A}$ uses block-coordinate descent to solve $(\mathcal{P})$ to a given *absolute* accuracy $\sigma \in (0, 1/3]$. It accepts as input an $x_0 \in B$ such that $\lambda(x_0) \le 1$, uses a $(\sigma/7, q)$-good block solver for a given relative block feasibility tolerance $q > 1$, and produces a solution x to $\lambda(x) \le \lambda^* + \sigma$, $x \in B$.

Algorithm $\mathcal{A}(f, x_0, \sigma, q)$:

Step 0. (Initialization.) Select $\delta = 6 \ln M$, $t = \sigma/7 \ln M$ (as defined by Lemma 1), and $\mu \in [\lambda(x_0), 1]$. Set x to x_0, and compute p from (2.2).

Step 1. (Block optimization.) For each block $k = 1, \ldots, K$, perform the following two steps:

Step 1.1. Call a $(\sigma/7, q)$-good block solver $\mathcal{BS}(f^k, p, \mu)$ to solve $(\mathcal{P}^k)$. Since $\lambda^* \le \lambda(x_0) \le \mu$, we have $B^k_\mu = \{\, x^k \in B^k,\ f^k(x^k) \le \mu e \,\} \neq \emptyset$, and the block must report an $\hat{x}^k \in B^k$ such that

$$\begin{aligned} &f^k(\hat{x}^k) \le \mu q e, \quad \text{and} \\ &p^T f^k(\hat{x}^k) \le \left(1 + \frac{\sigma}{7}\right) \min\{\, p^T f^k(x^k) \mid x^k \in B^k_\mu \,\} = \left(1 + \frac{\sigma}{7}\right) g_\mu(p). \end{aligned} \tag{4.1}$$

Step 1.2. For some step length $\tau^k \in (0, 1]$ (to be specified later), compute

$$\begin{aligned} \Delta^k_\Phi(\tau^k) &= \Phi_t(y(k, \tau^k)) - \Phi_t(x) \\ &= \ln \sum_{m=1}^{M} p_m \exp\left\{ \frac{f^k_m\big((1-\tau^k)x^k + \tau^k \hat{x}^k\big) - f^k_m(x^k)}{t} \right\}, \end{aligned}$$

where $y(k,\tau^k)$ is the point obtained from x by replacing its kth block coordinate x^k by $(1-\tau^k)x^k+\tau^k\hat{x}^k$.

Step 2. (Stopping criterion.) If the inequality $\sum_{k=1}^{K}\Delta_\Phi^k(\tau^k)\le -\sigma^2/28\mu q$ is not satisfied, report x and stop.

Step 3. (Coordination.) Select a k that minimizes $\Delta_\Phi^k(\tau^k)$ over $k=1,\dots,K$. Replace x by $y(k,\tau^k)$. For this new iterate x, compute a new p by (2.2). Go to Step 1.

This generic statement of Algorithm $\mathcal{A}$ can be modified and implemented in different ways. We shall introduce a randomized variant of this algorithm in §6, and discuss computational issues in §9.1. One of these is the normalization of computations in order to avoid the appearance of large numbers in the computation of p and $\Delta_\Phi^k(\tau^k)$.

In the next section we shall prove the correctness of Algorithm $\mathcal{A}$ and analyze its performance.

5. Analysis of Algorithm $\mathcal{A}$. We begin with the following technical lemma.

LEMMA 2. *Let* $E[f_1,\dots,f_M]=\sum_{m=1}^{M}p_mf_m$ *be the weighted average of* M *given numbers* $f_1,\dots,f_M$, *with weights* $p_m=\exp\{f_m/t\}\ /\ \sum_{i=1}^{M}\exp\{f_i/t\}$, $m=1,\dots,M$, *for some positive* t. *Then,*

$$\max\{f_1,\dots,f_M\}-E[f_1,\dots,f_M]\ \le\ t\ln M,$$

that is, for small t *the so-weighted average is close to the largest of the given numbers.*

Proof. Let $\max\{f_1,\dots,f_M\}=f_1$. Then,

$$\begin{aligned}\max\{f_1,\dots,f_M\}-E[f_1,\dots,f_M]&=\sum_{m=1}^{M}(f_1-f_m)p_m\\&=\sum_{m=1}^{M}(f_1-f_m)\exp\left\{-\frac{f_1-f_m}{t}\right\}\Bigg/\sum_{m=1}^{M}\exp\left\{-\frac{f_1-f_m}{t}\right\}\\&=t\left\{\frac{\sum_{m=2}^{M}\rho_m\exp\{-\rho_m\}}{1+\sum_{m=2}^{M}\exp\{-\rho_m\}}\right\},\end{aligned}$$

where $\rho_m=(f_1-f_m)/t\ge 0$. It remains to show that the quantity in the last large curly brackets above cannot exceed $\ln M$. To this end, suppose it exceeds some positive number θ, or equivalently,

$$\sum_{m=2}^{M}(\rho_m-\theta)\exp\{-\rho_m\}>\theta. \tag{5.1}$$

Consider the inequality $(\rho_m-\theta)\exp\{-\rho_m\}\le\exp\{-1-\theta\}$, which is valid for all $\theta>0$ and ρ_m: for a fixed θ, its left-hand side achieves its maximum at $\rho_m=\theta+1$. For this value of ρ_m, (5.1) implies that $\theta\exp\{1+\theta\}<M-1$, leading to a contradiction for $\theta=\ln M$. □

We are now ready to prove Algorithm $\mathcal{A}$ correct and state a bound on the number of its coordination steps for the special case of step length $\tau^1=\cdots=\tau^K=$ constant.

THEOREM 1. *For fixed* $\tau^k=\tau=\sigma t/6\mu q$, *and for an initial* $x_0\in B$ *such that* $\lambda(x_0)\le 1$, *Algorithm* $\mathcal{A}(f,x_0,\sigma,q)$ *halts in*

$$N=O\left(\frac{qK}{\sigma^2}\ln M\ln\left(2+\frac{\lambda(x_0)-\lambda^*}{\sigma}\right)\right) \tag{5.2}$$

coordination steps, with an $x \in B$ such that $\lambda(x) \leq \lambda^* + \sigma$.

Proof. Let x be the current iterate, and let $k \in \{1, \ldots, K\}$ be a block selected for processing in Step 1 of the algorithm. We shall first give an upper bound on the difference of the potential function

$$\Delta_\Phi^k(\tau) = \ln \sum_{m=1}^{M} p_m \exp\left\{ \frac{f_m^k((1-\tau)x^k + \tau\hat{x}^k) - f_m^k(x^k)}{t} \right\},$$

computed in Step 1.2 for the value $\tau^k = \tau$ given in the theorem. For convenience, denote $f_m^k = f_m^k(x^k)$, $\hat{f}_m^k = f_m^k(\hat{x}^k)$. By the convexity of $f_m^k(\cdot)$ we have

$$\begin{aligned} \Delta_\Phi^k(\tau) &\leq \ln \sum_{m=1}^{M} p_m \exp\left\{ -\frac{\tau}{t}(f_m^k - \hat{f}_m^k) \right\} \\ &\leq \sum_{m=1}^{M} p_m \left(\exp\left\{ -\frac{\tau}{t}(f_m^k - \hat{f}_m^k) \right\} - 1 \right). \end{aligned} \tag{5.3}$$

The last inequality follows from the observation that $\ln(u) \leq u - 1$ for all $u > 0$, and from the fact that $\sum_{m=1}^{M} p_m = 1$. We claim that

$$f_m^k, \hat{f}_m^k \in [0, \mu q]. \tag{5.4}$$

This is clearly the case for $\hat{f}_m^k$ by (4.1). It is also true initially for f_m^k since $\mu q \geq \mu \geq \lambda(x_0)$. At each subsequent iteration, the algorithm replaces x^k by a convex combination of x^k and $\hat{x}^k$, and the convexity of $f_m^k(\cdot)$ implies (5.4).

Now, using (5.4) and the assumptions $\tau^k = \tau = \sigma t / 6\mu q$, $\sigma \leq 1/3$, we have

$$\left| -\frac{\tau}{t}(f_m^k - \hat{f}_m^k) \right| \leq \frac{\sigma}{6} \leq \frac{1}{18}, \qquad m = 1, \ldots, M. \tag{5.5}$$

Since $\exp\{u\} - 1 \leq u + \frac{6}{11}u^2$ for all $|u| \leq \frac{1}{18}$, the inequality (5.3) can be further bounded as follows:

$$\Delta_\Phi^k(\tau) \leq -\frac{\tau}{t} \sum_{m=1}^{M} p_m (f_m^k - \hat{f}_m^k) + \frac{6\tau^2}{11t^2} \sum_{m=1}^{M} p_m (f_m^k - \hat{f}_m^k)^2.$$

In view of (5.4), we have $(f_m^k - \hat{f}_m^k)^2 \leq (f_m^k)^2 + (\hat{f}_m^k)^2 \leq \mu q (f_m^k + \hat{f}_m^k)$. Substituting this bound into the preceding inequality we obtain:

$$\Delta_\Phi^k(\tau) \leq -\frac{\tau}{t}\left(1 - \frac{6\mu q \tau}{11t}\right) p^T (f^k - \hat{f}^k) + \frac{12\mu q \tau^2}{11t^2} p^T \hat{f}^k. \tag{5.6}$$

In order to show that the best block k selected in Step 3 of the algorithm results in a substantial decrease in the potential, we give an upper bound on the average decrease over all blocks. To this end, we sum the upper bounds (5.6) for all k to obtain

$$\sum_{k=1}^{K} \Delta_\Phi^k(\tau) \leq -\tau\left(1 - \frac{6\mu q \tau}{11t}\right) D + \frac{12\mu q \tau^2}{11t^2} C, \tag{5.7}$$

where $D = D_1 - D_2$, $D_1 = p^T f/t$, $D_2 = p^T \hat{f}/t = C/t$, and $C = p^T \hat{f}$. The quantity C can easily be bounded from above by using both (4.1) and the duality relation ($\mathcal{D}$) of §1.2, as follows:

$$C \leq \left(1 + \frac{\sigma}{7}\right) \sum_{k=1}^{K} g_\mu^k(p) \leq \left(1 + \frac{\sigma}{7}\right) \lambda^* \leq \frac{22}{21}\lambda^* \leq \frac{22}{21}\lambda(x_0) \leq \frac{22}{21}. \tag{5.8}$$

Next, we derive a lower bound on $D = D_1 - D_2$ in terms of

$$\delta(x) = \Phi_t(x) - \Phi_t^*,$$

the absolute accuracy of minimization of the potential function[2] . To bound D_1 from below note that from Lemma 2, $D_1 = E[f_1, \ldots, f_M]/t \geq \max\{ f_1, \ldots, f_M \}/t - \ln M = \lambda(x)/t - \ln M$. By the second inequality of (2.3), $\lambda(x)/t \geq \Phi_t(x) - \ln M$. Hence, $D_1 \geq \Phi_t(x) - 2\ln M$. To bound $D_2 = C/t$ from above, use the second inequality of (5.8), $\lambda^* \leq 1$, property (2.4), and the parameter values of Step 0, to write

$$D_2 \leq \frac{\lambda^*}{t} + \frac{\lambda^*\sigma}{7t} \leq \frac{\lambda^*}{t} + \frac{\sigma}{7t} \leq \Phi_t^* + \frac{\sigma}{7t} = \Phi_t^* + \ln M.$$

Thus,

$$D = D_1 - D_2 \geq \Phi_t(x) - \Phi_t^* - 3\ln M = \delta(x) - 3\ln M. \tag{5.9}$$

By the stopping criterion in Step 2 and the choice of k in Step 3, the algorithm generates monotonically decreasing $\delta(x)$. By Lemma 1, if an iterate x enters the region $\delta(x) \leq \delta = 6\ln M$, then x minimizes $\lambda(x)$ over $x \in B$ to the desired absolute accuracy σ, albeit undetected when it occurs. For this reason, we consider the total number of iterations in two parts: N_1, the number of iterations needed to enter the region $\delta(x) \leq \delta$, and N_2, the number of additional iterations required to obtain the stopping criterion.

We shall first derive a bound for N_1. Suppose that $\delta(x) \geq \delta = 6\ln M$. By (5.9), this implies that $D \geq \delta(x)/2$. Substituting this bound, along with $C \leq 22/21$ into (5.7), results in

$$\begin{aligned}
\sum_{k=1}^{K} \Delta_\Phi^k(\tau) &\leq -\frac{\tau}{2}\left(1 - \frac{6\mu q\tau}{11t}\right)\delta(x) + \frac{8\mu q\tau^2}{7t^2} \\
&\leq -\delta(x)\left\{\frac{\tau}{2}\left(1 - \frac{6\mu q\tau}{11t}\right) - \frac{4\mu q\tau^2}{21t^2\ln M}\right\} \\
&< -\delta(x)\left\{\frac{\tau}{2} - \frac{3\mu q\tau^2}{2\sigma t}\right\}.
\end{aligned}$$

The last inequality follows from $\sigma \leq 1/3$ and from the initialization in Step 0 of the algorithm. Substituting our value of $\tau = \sigma t/6\mu q$, which maximizes the expression in the last pair of curly brackets, we obtain

$$\sum_{k=1}^{K} \Delta_\Phi^k(\tau) \leq -\left(\frac{c\sigma^2}{\mu q \ln M}\right)\delta(x) \tag{5.10}$$

[2] For linear functions f_m^k and for exact block solvers, the quantity D provides the decrease in the linear approximation to the potential function at x. By the convexity of $\Phi_t(\cdot)$, this implies $D \geq \delta(x)$.

with $c = 1/168$. In particular, for $\delta(x) \geq \delta$ we have the inequality given in Step 2, whose negation is the stopping criterion[3]. This proves the correctness of Algorithm $\mathcal{A}$. Furthermore, (5.10) and the choice of the best k in Step 3 of the algorithm yield the recurrence

$$\delta(x_l) \geq \delta \quad \Rightarrow \quad \delta(x_{l+1}) \leq \left(1 - \frac{c\sigma^2}{\mu q K \ln M}\right) \delta(x_l),$$

where x_l is the lth iterate of the algorithm. Hence, the number of iterations N_1 of the algorithm sufficient for an x to enter the region $\delta(x) \leq \delta$ can be bounded by

$$N_1 \leq \left\lceil \frac{\mu q K \ln M}{c\sigma^2} \ln \frac{\delta(x_0)}{\delta} \right\rceil.$$

In view of (2.3)–(2.4),

$$\delta(x_0) \leq \frac{\lambda(x_0) - \lambda^*}{t} + \ln M,$$

which upon substitution of our values for t and δ, gives

$$\ln \frac{\delta(x_0)}{\delta} \leq \ln \left(\frac{1}{6} + \frac{7}{6} \frac{\lambda(x_0) - \lambda^*}{\sigma}\right) = O\left(\ln\left(2 + \frac{\lambda(x_0) - \lambda^*}{\sigma}\right)\right). \tag{5.11}$$

Hence, N_1 is bounded by

$$N_1 = O\left(\frac{\mu q K}{\sigma^2} \ln M \ln\left(2 + \frac{\lambda(x_0) - \lambda^*}{\sigma}\right)\right).$$

It remains to bound N_2. By the stopping criterion and the choice of k in Step 3, the potential decreases by at least $\Delta = \sigma^2/28\mu q K$ at each iteration. Therefore,

$$N_2 \leq \left\lceil \frac{\delta}{\Delta} \right\rceil = O\left(\frac{\mu q K \ln M}{\sigma^2}\right).$$

Since $\mu \leq 1$, the above bounds for N_1 and N_2 prove the theorem. □

6. A randomized variant of Algorithm $\mathcal{A}$. Each iteration of Algorithm $\mathcal{A}$ solves K independent block problems and then selects the best one of these solutions to compute a new iterate x and the corresponding price vector p. A natural question is whether solving fewer block problems per step might provide comparable performance. In this section, we shall state and analyze a *randomized variant* of this algorithm, *Algorithm* $\mathcal{R}$, which solves two block problems per step, on the average. As we shall see, the expected number of iterations of this variant is within a constant factor (of about two) of the bound given by Theorem 1 for the deterministic Algorithm $\mathcal{A}$. Thus, the randomized algorithm is advantageous for problems with a large number of blocks.

Algorithm $\mathcal{R}(f, x_0, \sigma, q)$:

Step R0. (Initialization.) Same as Step 0 of Algorithm $\mathcal{A}$.

Step R1. (Randomization.) Flip a $K+1$-sided fair coin giving the outcome $k \in \{0, 1, \ldots, K\}$. If $k = 0$, go to Step *R3*.

---

[3] For the linear case, where $D \geq \delta(x)$, the optimal choice of the step length τ is approximately twice the value used in Theorem 1. Consequently, the value of c (the guaranteed decrease in the potential in the stopping criterion of Step 2) increases and the iteration bound decreases by about a factor of four.

Step R2. (Process random block k.) Perform Steps 1.1–1.2 of Algorithm $\mathcal{A}$ for the kth block, which compute $\Delta_\Phi^k(\tau^k)$. If $\Delta_\Phi^k(\tau^k) < 0$, replace x by $y(k, \tau^k)$ (this guarantees the monotonicity of potential function values), and compute a new p by (2.2). Go to Step $R1$.

Step R3. (Process all blocks.) Perform one complete iteration (Steps 1–3) of Algorithm $\mathcal{A}$. If Algorithm $\mathcal{A}$ does not halt, go to Step $R1$.

Our analysis of Algorithm $\mathcal{R}$ assumes the uniform step size used in Theorem 1.

THEOREM 2. *For fixed $\tau^k = \tau = \sigma t/6\mu q$, and for an initial $x_0 \in B$ such that $\lambda(x_0) \leq 1$, Algorithm $\mathcal{R}(f, x_0, \sigma, q)$ halts in expected*

$$E[N] = O\left(\frac{qK}{\sigma^2} \ln M \ln\left(2 + \frac{\lambda(x_0) - \lambda^*}{\sigma}\right)\right)$$

coordination steps, with an $x \in B$ such that $\lambda(x) \leq \lambda^* + \sigma$.

Proof. Algorithm $\mathcal{R}$ halts only if $k = 0$ and the stopping criterion of Algorithm $\mathcal{A}$ is satisfied. That is, $\mathcal{R}$ halts if the current outcome in Step $R1$ is $k = 0$ and $\sum_{k=1}^K \Delta_\Phi^k(\tau^k) > -c_1$, where $c_1 = \sigma^2/28\mu q$. In particular, $\mathcal{R}$ halts if $k = 0$ and $\sum_{k=1}^K \min\{0, \Delta_\Phi^k(\tau^k)\} > -c_1$. To simplify the proof, we shall replace the stopping criterion in Step 2 of Algorithm $\mathcal{A}$ by this stronger condition. This can only increase the expected number of iterations.

Algorithm $\mathcal{R}$ corresponds to a stochastic process which can be modeled as a *random descent* P in an infinite $(K+1)$-ary directed rooted tree G as follows. The root of G corresponds to the initial iterate $x_0 \in B$, and the $(K+1)^l$ vertices in the lth level of G correspond to all possible lth iterates $x_l \in B$ of Algorithm $\mathcal{R}$. The adjacency $A(x_l)$ consists of $K+1$ arcs from x_l to its $K+1$ children, each of which represents one of the $K+1$ outcomes of the coin flip in Step $R1$. For each vertex x_l, the leftmost arc $e_{x_l} \in A(x_l)$ models the execution of one iteration of Algorithm $\mathcal{A}$, i.e., the outcome $k = 0$ in Step $R1$.

Assign the nonnegative weight $\delta(x_l) = \Phi_t(x_l) - \Phi^*$ to each vertex x_l of G. Since Algorithm $\mathcal{R}$ generates monotonically decreasing values of the potential, the sequence of vertex weights encountered by a path P from the root to any vertex of G is nonincreasing. Also assign the nonnegative weight $w(x_l, x_{l+1,k}) = \delta(x_l) - \delta(x_{l+1,k}) = \Phi_t(x_l) - \Phi_t(x_{l+1,k})$ to each arc $(x_l, x_{l+1,k})$, $k = 0, 1, \ldots, K$, of G. We shall call the leftmost arc in the $A(x_l)$ of any vertex x_l a *stopping arc* if the total weight of the remaining K arcs, $w(A(x_l)\backslash\{e_{x_l}\})$, is less than c_1. This corresponds to the modified stopping criterion stated in the first paragraph of this proof.

Imagine a stroller who starts at the root of G with an amount of cash $\delta(x_0)$ and who descends the tree along a path P, randomly selecting an arc $e \in A(x_l)$ to traverse at each vertex x_l he visits. For every arc e the stroller traverses, he pays a "toll" $w(e)$. Since for any N, the weight $w_N(P)$ of an arbitrary descent P with N arcs from the root cannot exceed $\delta(x_0)$, the stroller can never go into debt and his walk will not terminate. Now, modify the weight of all stopping arcs to be a large number (exceeding $\delta(x_0)$ and c_1). Then, the first time the stroller traverses a stopping arc he goes into debt, and conversely. This means that in this modified tree model of Algorithm $\mathcal{R}$, the expected number of iterations of $\mathcal{R}$ is equal to the expected number of steps $E[N]$ needed to bankrupt the stroller.

Represent the random variable N as the sum of two random variables: N_1, the time (number of arcs traversed) when the stroller reaches the "level of poverty" $2\delta = 12 \ln M$, and N_2, the additional time to bankrupt the stroller. In order to bound the

expectations $E[N_1]$ and $E[N_2]$, observe that the (redefined) arc-weight function w has the following two properties:

(1) $w(A(x_l)) > c_1 = \sigma^2/28\mu q$ for any vertex x_l of G, and

(2) $\delta(x_l) \geq \delta = 6 \ln M \Rightarrow w(A(x_l)) \geq c_2 \delta(x_l)$, where $c_2 = \sigma^2/168\mu q \ln M$. This inequality corresponds to (5.10) and can be shown[4] exactly as in the proof of Theorem 1.

To proceed we need the following lemma, whose proof we shall postpone.

LEMMA 3. *Let G be a d-ary tree with a nonnegative arc-weight function u such that $u(A(v)) \geq C$ for each vertex of $v \in G$, where $C > 0$ is a constant. Let $U \geq 0$ be a given threshold weight. For a random descent P in G, let $T = \min\{ t \mid u_t(P) \geq U \}$, the first instant when the weight $u_t(P)$ of the first t arcs in P reaches U. Then,*

$$E[T] \leq d(1 + U/C).$$

Using this lemma with $d = K + 1$, $U = 2\delta$, and $C = c_1 = \sigma^2/28\mu q$ from property (1) and the fact that $\mu \leq 1$, we immediately obtain the bound

$$E[N_2] \leq (K+1)\left(1 + \frac{2\delta}{c_1}\right) = O\left(\frac{qK \ln M}{\sigma^2}\right).$$

To bound $E[N_1]$, suppose that at some instant we have $\delta(x_l) \geq 2\delta$. Then, by property (2) and by Lemma 3 with $d = K + 1$, $U = \delta(x_l)/2$, and $C = c_2\delta(x_l)/2$, the expected number of iterations needed to halve $\delta(x_l)$ does not exceed

$$(K+1)\left(1 + \frac{1}{c_2}\right) = O\left(\frac{qK \ln M}{\sigma^2}\right),$$

which results in the bound:

$$E[N_1] = O\left(\frac{qK \ln M}{\sigma^2} \left\lceil \ln \frac{\delta(x_0)}{\delta} \right\rceil\right).$$

The bounds for $E[N_1]$ and $E[N_2]$, along with (5.11), prove the theorem. □

We now give a simple inductive proof of Lemma 3. Samorodnitsky [23] gave an earlier proof of this lemma using the optional stopping theorem for martingales.

Proof of Lemma 3. Without loss of generality we can assume that $u(A(v)) = C$ for each vertex of $v \in G$. We first prove the lemma for integer-valued u, U, and C. Let v_0 be the root of G, and denote the weights $u(A(v_0))$ by $u_1, \ldots, u_d$. The expectation $E[T]$, viewed as a function $E_G(U)$ of G and U, satisfies the recurrent inequality $E_G(U) \leq 1 + (1/d)\sum_{k=1}^{d} E_{G_k}([U - u_k]^+)$, where G_k denotes the subtree of G rooted at the kth child of the root, and $[x]^+ = \max\{0, x\}$. Taking suprema over the right-hand and then the left-hand side of this inequality yields the recurrence

$$E(U) \leq \max\left\{ 1 + \frac{1}{d}\sum_{k=1}^{d} E([U - u_k]^+) \right\}, \tag{6.1}$$

where $E(U)$ is the supremum of $E_G(U)$ over all G satisfying the assumption $u(A(v)) = C$ for each $v \in G$, and the maximum is taken over all d-tuples of integers $\{u_1, \ldots, u_d\} \geq$

[4] This is a necessary exercise because the sequence of $\lambda(x)$ is not monotonic in general, and thus the inequality $\lambda(x_l) \leq 1$ is not assured for any iteration of Algorithm $\mathcal{R}$. The proof of (5.10) in Theorem 1 is valid because it uses the assumption $\lambda(x) \leq 1$ only for the very first iterate x_0.

0 that sum up to C. Observe that $E(0) = 0$, and that $E(U)$ is a finite nondecreasing function of $U = 0, 1, \ldots$. Now use (6.1) to prove by induction on U that $E(U) \leq d(1 + U/C)$, which coincides with the claim of the lemma for the integer case.

For a given $U = 1, 2, \ldots$, the maximum in (6.1) is attained for some d-tuple $\{u_1, \ldots, u_d\}$. If $u_k \geq U$ for some $k \in \{1, \ldots, d\}$, then one of summands in (6.1) is zero. Since $E([U - u_k]^+) \leq E(U)$, we have $E(U) \leq 1 + 0 + \frac{d-1}{d} E(U)$, or $E(U) \leq d$. Suppose now that $u_k < U$ for all $1 \leq k \leq d$, and that there are $0 \leq p < d$ zero u_k's, say, $u_1 = \cdots = u_p = 0$. Rewriting (6.1) as $\left(1 - \frac{p}{d}\right) E(U) \leq 1 + \frac{1}{d} \sum_{k=p+1}^{d} E(U - u_k)$ and using the induction hypothesis $E(U - u_k) \leq d(1 + (U - u_k)/C)$ and $\sum_{k=p+1}^{d} u_k = C$ completes the proof for the integer case, and consequently for rational-valued u, C, and U with common denominator $Q > 0$. The real-valued case follows by approximating U from above and u at each arc from below by nonnegative rational numbers with denominator $Q \to \infty$. □

7. A ternary search procedure. The algorithms presented in §§4 and 6 solve problem $(\mathcal{P})$ to an absolute accuracy σ. In this section we shall state a *ternary search* procedure $\mathcal{TS}$ which computes the optimal value λ^* to our original problem $(\mathcal{P}_\varepsilon)$ for a given relative accuracy $\varepsilon > 0$, assuming that an initial approximation $x_0 \in B$ is given. We shall no longer assume that $\lambda(x_0) \leq 1$.

A deterministic implementation of this procedure requires an initial point $x_0 \in B$, it uses Algorithm $\mathcal{A}$ as a subroutine, and maintains a lower bound $\underline{\lambda}$ and an upper bound $\bar{\lambda}$ to the optimal value λ^*. Its randomized variant uses Algorithm $\mathcal{R}$ instead of Algorithm $\mathcal{A}$.

Algorithm $\mathcal{TS}(x_0, \varepsilon, q)$:

Step 0. Set $\underline{\lambda} = 0$, $x = x_0$, and $\bar{\lambda} = \lambda(x)$. Then repeat the following three steps until the stopping criterion $\lambda(x) = \bar{\lambda} \leq (1 + \varepsilon)\underline{\lambda}$ is satisfied:

Step 1. Set

$$\nu = \frac{1}{\bar{\lambda} + 2\underline{\lambda}}, \qquad \sigma = \frac{1}{3}\left(\frac{\bar{\lambda} - \underline{\lambda}}{\bar{\lambda} + 2\underline{\lambda}}\right) \tag{7.1}$$

and call Algorithm $\mathcal{A}(\nu f, x, \sigma, q)$. (Theorem 1 applies here since $\sigma \leq 1/3$ and $\nu\lambda(x) = \nu\bar{\lambda} \leq 1$.) Algorithm $\mathcal{A}$ provides a point $y \in B$ such that $\nu\lambda(y) \leq \nu\lambda^* + \sigma$, i.e.,

$$\lambda(y) \leq \lambda^* + (\bar{\lambda} - \underline{\lambda})/3. \tag{7.2}$$

Step 2. If $\lambda(y) \leq \lambda_2 = (1/3)\underline{\lambda} + (2/3)\bar{\lambda}$, replace x by y and the upper bound $\bar{\lambda}$ by $\lambda(y)$.

Step 3. If $\lambda_2 < \lambda(y)$, then $\lambda^* > \lambda_1 = (2/3)\underline{\lambda} + (1/3)\bar{\lambda}$ by (7.2). Replace the lower bound $\underline{\lambda}$ by λ_1.

Procedure $\mathcal{TS}$ is correct: it generates a sequence of intervals $[\underline{\lambda}, \lambda(x)]$ containing λ^*, and each interval shrinks by at least a factor of 2/3 from one iteration of $\mathcal{TS}$ to the next.

In order to bound the complexity of TS, observe that for each iteration,

$$\frac{\nu\lambda(x) - \nu\lambda^*}{\sigma} = \frac{\nu\bar{\lambda} - \nu\lambda^*}{\sigma} \leq \frac{\nu\bar{\lambda} - \nu\underline{\lambda}}{\sigma} = 3.$$

Thus, by (5.2), the number N of coordination steps of Algorithm $\mathcal{A}$ for each iteration of $\mathcal{TS}$ is bounded by

$$N = O\left(\frac{qK \ln M}{\sigma^2}\right) = O\left(qK \ln M \left(\frac{\bar{\lambda} + 2\underline{\lambda}}{\bar{\lambda} - \underline{\lambda}}\right)^2\right),$$

where $\underline{\lambda}$ and $\bar{\lambda}$ are the bounds for the current iteration. We claim that

$$\sum \left(\frac{\bar{\lambda}+2\underline{\lambda}}{\bar{\lambda}-\underline{\lambda}}\right)^2 = O\left(\varepsilon^{-2} + \ln\frac{\lambda(x_0)}{\lambda^*}\right), \tag{7.3}$$

with the sum taken over all iterations of TS. To see this, partition the iterations performed by $\mathcal{TS}$ into two parts: iterations for which $\underline{\lambda} = 0$, and the subsequent iterations for which $\underline{\lambda} > 0$. Each iteration in the first part causes the upper bound $\bar{\lambda} = \lambda(x)$ to decrease by at least a factor of $2/3$. Hence,

$$\sum_{\underline{\lambda}=0} \left(\frac{\bar{\lambda}+2\underline{\lambda}}{\bar{\lambda}-\underline{\lambda}}\right)^2 = O\left(\ln\frac{\lambda(x_0)}{\lambda^*}\right).$$

For each iteration in the second part, the positivity of $\underline{\lambda}$ implies that $\bar{\lambda} \le 3\underline{\lambda}$. Consequently,

$$\frac{1}{3}\left(\frac{\underline{\lambda}}{\bar{\lambda}-\underline{\lambda}}\right) \le \frac{1}{5}\left(\frac{\bar{\lambda}+2\underline{\lambda}}{\bar{\lambda}-\underline{\lambda}}\right) \le \frac{\underline{\lambda}}{\bar{\lambda}-\underline{\lambda}} \le \frac{\lambda^*}{\bar{\lambda}-\underline{\lambda}} \le \frac{3\underline{\lambda}}{\bar{\lambda}-\underline{\lambda}}, \tag{7.4}$$

that is, $(\bar{\lambda}+2\underline{\lambda})/(\bar{\lambda}-\underline{\lambda})$ and $\underline{\lambda}/(\bar{\lambda}-\underline{\lambda})$ are of the same order as $\lambda^*/(\bar{\lambda}-\underline{\lambda})$, the reciprocal of the current relative error.

Since at each iteration the gap $\bar{\lambda}-\underline{\lambda}$ decreases by at least a factor of $2/3$, by the stopping criterion of $\mathcal{TS}$ we have

$$\sum_{\underline{\lambda}>0} \left(\frac{\bar{\lambda}+2\underline{\lambda}}{\bar{\lambda}-\underline{\lambda}}\right)^2 = O\left(\varepsilon^{-2}\left[1+\left(\frac{2}{3}\right)^2+\left(\frac{2}{3}\right)^4+\cdots\right]\right) = O(\varepsilon^{-2}),$$

where ε is the desired (final) relative accuracy of the approximation. This proves (7.3).

We have thus shown that the total number of coordination steps $N(\varepsilon)$ required to solve $x \in B$, $\lambda(x) \le (1+\varepsilon)\lambda^*$ is bounded by

$$N(\varepsilon) = O\left(qK \ln M\left(\varepsilon^{-2} + \ln\frac{\lambda(x_0)}{\lambda^*}\right)\right). \tag{7.5}$$

Clearly, this bound also holds for the expected number of iterations for the randomized variant of TS.

Note that by the choice of σ in (7.1) and the inequalities (7.4), Algorithm $\mathcal{TS}(x_0, \varepsilon, q)$ calls an (ε', q)-good block solver via Algorithm $\mathcal{A}$, where $\varepsilon' = \varepsilon/(7\cdot 3\cdot 5) \approx 0.01\varepsilon$. That is, the relative optimization accuracy demanded of the block solver differs only by a constant factor from the relative tolerance prescribed for $(\mathcal{P}_\varepsilon)$. This is true for the randomized variant of $\mathcal{TS}$ as well.

8. The overall approximation scheme and its complexity. The ternary search procedure $\mathcal{TS}$ of the previous section solves the original problem $(\mathcal{P}_\varepsilon)$ provided that an arbitrary $x_0 \in B$ is known. The number of block-solver calls $\mathcal{TS}$ generates depends only logarithmically on the relative accuracy of the available approximation. Although this is not an overly restrictive assumption in practice, it can be easily removed by using a simple *greedy initialization procedure* $\mathcal{GP}$ which we shall describe below. Thus, our overall scheme for $(\mathcal{P}_\varepsilon)$ will consist of $\mathcal{GP}$ followed by TS. We shall

denote the resulting deterministic and randomized variants of this scheme by $\mathcal{DS}$ and $\mathcal{RS}$, respectively.

The greedy initialization procedure $\mathcal{GP}$ operates separately on each block k as follows. Initially, set $p = e/M \in P$ and $\mu = \infty$. Then, call a $(1, q)$-good block solver to compute a solution $\hat{x}^k$ to problem $(\mathcal{P}^k)$:

$$p^T f^k(\hat{x}^k) \leq 2 \min\{ p^T f^k(x^k) \mid x^k \in B^k_\mu \}. \tag{8.1}$$

(The block solver must provide such a solution since initially $\mu = \infty$.) Clearly, the above minimum does not exceed $\lambda^*_k = \min\{ \lambda \mid f^k(x^k) \leq \lambda e, \quad x^k \in B^k \}$. This fact, and the nonnegativity of the $f^k(\cdot)$, imply that $f^k(\hat{x}^k) \leq \mu e$ with $\mu \leq 2M\lambda^*_k$. Now, halve μ and repeat the above step for any p until the block solver reports (possibly incorrectly) that $B^k_{\mu q} = \emptyset$ for some value of $\mu = \mu_1$. Recalling our assumptions on the performance of the block solver in §3, we conclude that for $\mu_2 = 2q\mu_1$, the set $B^k_{\mu_2} \neq \emptyset$. Now, use the block solver for $\mu = \mu_2$ to compute an $\hat{x}^k$ to (8.1) and save this solution as x^k_0. It is easy to see that $f^k(x^k_0) \leq 2q^2\lambda^*_k e$.

The total number of block-solver calls issued by $\mathcal{GP}$ is $O(\ln M + \ln q)$ per block. Since $f^k(x^k_0) \leq 2q^2\lambda^*_k e$ for each x^k_0 computed by $\mathcal{GP}$, we have $f(x_0) \leq 2q^2 \sum_{k=1}^K \lambda^*_k e \leq 2Kq^2\lambda^* e$. Thus $\lambda(x_0)/\lambda^* \leq 2Kq^2$. Incorporating these bounds into (7.5) of §7, we obtain our final bounds for the overall algorithm.

THEOREM 3. *The deterministic approximation scheme $\mathcal{DS}$ solves problem (P) with nonnegative-valued block-separable convex coupling functions $f(x)$ to a given relative accuracy $\varepsilon \in (0, 1]$ in*

$$N(\varepsilon) = O\left(qK \ln M \left(\varepsilon^{-2} + \ln qK\right)\right)$$

coordination steps, each generating K parallel calls to $(const \cdot \varepsilon, q)$-good block solvers. This expression also bounds the expected number of calls to such block solvers for the randomized variant $\mathcal{RS}$.

Thus, for $q = 1 + O(\varepsilon)$, or even for $q = O(1)$, Theorem 3 provides the bound (1.4) claimed in the Introduction. An application to additively-separable functions f^k_m, which includes the linear case, is stated as Corollary 1 in §9.2.

9. Extensions and applications. We have shown how to render the block-coordinate descent strategy efficient for computing an approximate solution to the nonlinear program $(\mathcal{P})$. We have presented a deterministic and a randomized algorithm, both of which are fully polynomial in the number of blocks they solve and in ε^{-1}. These algorithms result in fully polynomial-time approximation schemes for some special cases of linear programming, such as ordinary and generalized flow networks with additional constraints, and generalized multicommodity concurrent flows. Our analysis suggests that the algorithms developed here, as well as those in [21], [25], [14], [16], and [18] can have a combinatorial flavor to the extent that some applications result in block problems amenable to efficient combinatorial methods. We shall illustrate this point with several applications later in this section. Since the number of coupling constraints M enters the complexity bounds only logarithmically, we expect that additional applications will be found.

9.1. Computational issues. As we have seen, Algorithms $\mathcal{A}$ and $\mathcal{R}$ of §§4 and 6 are the key ingredients of the overall method for problem $(\mathcal{P})$. The generic description of these algorithms suggests a number of straightforward enhancements. We enumerate some of these below.

Stopping criteria. There are at least two additional stopping criteria that can be incorporated into Algorithm $\mathcal{A}$ and into the ternary search procedure $\mathcal{TS}$ to affect an earlier termination. First, by (5.9), Algorithm $\mathcal{A}$ can be terminated whenever $D \leq 3 \ln M$ or equivalently whenever the approximate duality gap $p^T(f(x) - f(\hat{x})) \leq \frac{3}{7}\sigma$. Second, since Algorithm $\mathcal{A}$ is used by $\mathcal{TS}$ as a subroutine, its branching rule $\lambda(y) \leq \lambda_2 = \frac{1}{3}\lambda + \frac{2}{3}\bar{\lambda}$ can also be used to terminate $\mathcal{A}$. The same criteria also benefit Algorithm $\mathcal{R}$ since it uses $\mathcal{A}$ as a subroutine.

Normalization and accuracy of computations. At each iteration of Algorithms $\mathcal{A}$ and $\mathcal{R}$ it is necessary to recompute the price vector p by (2.2), which can be written in the normalized form [21]:

$$p_m = \frac{\exp\{(f_m(x) - \lambda(x))/t\}}{\sum_{i=1}^{M} \exp\{(f_i(x) - \lambda(x))/t\}}, \qquad m = 1, \ldots, M,$$

where the exponents are nonpositive. Furthermore, the exponents in the expression for $\Delta_\Phi^k(\tau^k)$ used in Step 1.2 of Algorithm $\mathcal{A}$ have been shown to be small by the inequalities (5.3) and (5.5) in the proof of Theorem 1. Hence, the generic form of Algorithm $\mathcal{A}$ can be implemented in a way that avoids having to deal with large numbers.

It can also be shown that if the values of the $p_m \in [0, 1]$ are computed to an absolute accuracy of $l = O(\ln(KMq/\varepsilon))$ bits and are then passed to an approximate block solver of §3, the iteration bounds derived earlier remain valid. We omit the proof, which requires a minor modification of the initialization procedure of §8 to freeze the greedy solutions x_0^k for those blocks that have $\lambda_k(x_0^k)$ of order 2^{-l} relative to the $\max_k \lambda_k(x_0^k)$. Further analysis of the accuracy of computation also requires specific knowledge about the functions f^k and blocks B^k, their representation, and the particular method used by the block solvers. For example, in the special case of matrix games, such an analysis in the bit model of computation is provided in [9].

Line search. Suppose that instead of the quantity $\Delta_\Phi^k(\tau)$ defined in Step 1.2 of Algorithm $\mathcal{A}$, we use throughout the quantity

$$\Delta^k\left(\frac{\tau^k}{t}\right) = \ln \sum_{m=1}^{M} p_m \exp\left\{-\frac{\tau^k}{t}\left(f_m^k - \hat{f}_m^k\right)\right\},$$

which is an upper bound on $\Delta_\Phi^k(\tau)$ as stated in (5.3). (These two quantities coincide when $f^k(\cdot)$ are linear.) Theorems 1, 2 and 3 remain valid for this change because our entire analysis uses this approximation instead of $\Delta_\Phi^k(\tau)$. They also remain valid if the step length is selected independently for each block by performing a line search for $\Delta^k(\tau^k/t)$, instead of using the same constant step length for all blocks as in our previous analysis. This may improve performance in practice, particularly when used with standard heuristics for coordinate descent methods.

K-search. Potentially larger improvements can be attained by replacing the K independent line searches for Algorithm $\mathcal{A}$ by a single K-dimensional approximate minimization of the function

$$\Delta\left(\frac{\tau^1}{t}, \ldots, \frac{\tau^K}{t}\right) = \ln \sum_{m=1}^{M} p_m \exp\left\{-\sum_{k=1}^{K} \frac{\tau^k}{t}(f_m^k - \hat{f}_m^k)\right\},$$

which bounds from above the decrease

$$\Phi_t(x^1(1-\tau^1) + \tau^1\hat{x}^1, \ldots, x^K(1-\tau^K) + \tau^K\hat{x}^K) - \Phi_t(x^1, \ldots, x^K)$$

in the potential function (see (5.3)). In order to avoid large exponents, a normalization similar to the one described for p above can be used for this multidimensional search in the variables $\tau^1/t, \ldots, \tau^K/t$. In fact, the minimization of Δ is a simple geometric programming problem with box constraints on the variables, which can be solved efficiently in practice to a fixed accuracy of order $\sigma^2/\mu q$ for moderate values of K. This idea, albeit for a different kind of potential function, was proposed by Schultz and Meyer [24] who used it to solve large multicommodity flow problems with $K = 11$. Multidimensional search can be performed for the randomized algorithm $\mathcal{R}$ as well, by randomly selecting a small subset of blocks $\mathcal{K} \subseteq \{1, \ldots, K\}$.

Constraint relaxation for block solvers. The practical efficiency of block solvers that operate on the sets B_μ^k can often be improved by omitting those constraints $f_m^k(x^k) \leq \mu$ which either correspond to relatively small values p_m or remain inactive for a block problem over a number of coordination steps. It is possible to devise adaptive strategies which periodically reconstitute the set of constraints $f_m^k(x^k) \leq \mu$ included in B_μ^k, similar to those described by Grigoriadis and White [11]. Such heuristic strategies were found to be very effective for linear multicommodity problems with weighted coupling constraints on arc flows [11].

Relaxation of the threshold q. The bounds of Theorem 3 suggest that there is a certain amount of trade-off between the number of coordination steps and the relative feasibility tolerance q prescribed to the block solvers. The larger the value of q, the easier the task of the block solver but the larger the bound on the number of coordination steps. Perhaps the simplest example of such a trade-off is the device of box-relaxed block problems discussed immediately below. The next two applications also exploit this idea.

9.2. Applications. We conclude with a few applications that illustrate the use of the algorithms presented in this paper.

Additively separable functions and box relaxation. As mentioned in §1.1, the case of $(\mathcal{P})$ with additively separable or linear functions $f_m^k(x^k)$ is of considerable practical importance. That is,

$$f_m^k(x^k) = \sum_{j \in J_m^k} f_m^{kj}(x^{kj}),$$

where f_m^{kj} are nonnegative convex functions and J_m^k is the set of indices of those elements of x^k that enter the function $f_m^k(x^k)$. We shall denote the largest number of block elements that enter any coupling constraint of $(\mathcal{P})$ by $S = \max_{k,m} |J_m^k|$, and refer to this number as the *block coupling degree* of $(\mathcal{P})$. S is the maximum dimension of the x^k in the worst case, but in many problems it is a small constant, frequently one.

For additively separable functions the block problems can be replaced by the problems of approximately minimizing $p^T f^k(x^k)$ over the *box-relaxed* sets $BB_\mu^k = B^k \cap Box_\mu^k$, where $Box_\mu^k = \{\, x^k \mid f_m^{kj}(x^{kj}) \leq \mu, \;\; j \in J_m^k, \;\; m = 1, \ldots, M \,\}$. This can be much simpler than minimizing the same function over the sets B_μ^k. The resulting *box-relaxed solution* $\hat{x}^k \in BB_\mu^k$ can then be used as an approximate solution to the original block problems $(\mathcal{P}^k)$. This idea can be formalized by defining an (ε', q)-*good box-block solver* $\mathcal{BBS}(f^k, p, \mu)$ for $(\mathcal{P}^k)$ as follows: in the definition of $\mathcal{BS}$ in §3, replace (3.1) by $\hat{x}^k \in BB_{\mu q}^k$, and leave (3.2) intact. Also, assume that $\mathcal{BBS}$ succeeds whenever $B_\mu^k \neq \emptyset$, it correctly reports all instances of $BB_{\mu q}^k = \emptyset$, but behaves unreliably whenever $B_\mu^k = \emptyset$ and $BB_{\mu q}^k \neq \emptyset$. (In short, replace $B_{\mu q}^k$, but not B_μ^k, by $BB_{\mu q}^k$.)

By the nonnegativity of the functions $f^{kj}(x^{kj})$, we have $B^k_\mu \subseteq BB^k_\mu \subseteq B^k_{\mu S}$ for all $\mu > 0$, where S is the block coupling degree of $(\mathcal{P})$. This implies that any (ε', q)-good box-block solver $\mathcal{BBS}(f^k, p, \mu)$ can be used as an (ε', qS)-good block solver $\mathcal{BS}(f^k, p, \mu)$. Theorem 3 now implies the following claim.

COROLLARY 1. *If in addition to the assumptions of Theorem* 3, *the coupling functions f^k_m are additively separable, the bounds of Theorem* 3 *are*

$$N(\varepsilon) = O\left(qSK \ln M \left(\varepsilon^{-2} + \ln qSK\right)\right)$$

coordination steps using $(const \cdot \varepsilon, q)$-good box-block solvers.

Concurrent flows with uniform capacities [21]. Consider a communication network having n-vertices and M-arcs representing channels of uniform capacity λ. There are $K \leq n$ commodities which will share this network. A commodity k is associated with a supply $b^k > 0$ at a vertex v^k and possibly $n-1$ demands distributed over the rest of the vertices, to be satisfied exactly. The problem is to assign the minimum capacity λ to each arc of the network so that all commodity flows can be accommodated. This is a linear problem $(\mathcal{P})$ with M resource constraints of coupling degree $S = 1$, and K blocks B^k, each of which consists of flow conservation and nonnegativity constraints. The maximum vertex degree d in communication networks is typically small. For instance, $d = \log n$ in a hypercube.

The solution of this problem can be approached in many ways. On one hand, if $q = O(1)$, the solution to a relative accuracy ε can be obtained in expected $K \ln n(\varepsilon^{-2} + \ln K)$ uniformly capacitated minimum-cost flow computations. If in addition there is only one nonzero demand b^k for a commodity k, the corresponding block problem is equivalent to a single-pair shortest path computation whenever $b' = b^k/\mu < 1$. Otherwise $(b' \geq 1)$, a $(0, 2)$-good block solver is easily constructed by solving the block problem with supply and demand of $b'' = \mu\lfloor b' \rfloor$ and by multiplying the resulting solution by $b'/\lfloor b' \rfloor$. Using a standard reduction, the problem with b'' is equivalent to an assignment problem in $O(M)$ vertices and edges, which can be solved in $O(M^{3/2} \ln(n\varepsilon^{-1}))$ time, as stated in Theorem 5.4 in [16]. Thus the algorithm can be implemented to run in expected $O(KM^{3/2} \ln n \ln(n\varepsilon^{-1})(\varepsilon^{-2} + \ln K))$ time, which is faster than the bound given in [16]. On the other hand, setting $q = d$ effectively removes the capacity constraints in $B^k_{\mu q}$. This simplifies the minimum-cost problems mentioned above to single-source shortest-path problems. Thus, the problem can also be solved in expected $O(dK \ln n(\varepsilon^{-2} + \ln dK)(M + n \ln n))$ time, even for the case with multiple demands per commodity. For a sufficiently small ε, the ratio of the latter time to the former is at most $d/\sqrt{M}$, which shows that this trade-off is advantageous for small d. These two extreme cases can be combined to create data-dependent hybrid algorithms which adaptively select $q \in [1, d]$ and solve some block problems as assignment and some others as shortest-path problems. The solution of such block problems can be further parallelized. For example, for a fixed ε and up to logarithmic factors, the second of the above bounds can be reduced to $O(dK)$ parallel time.

Generalized concurrent flow problems with side constraints. We define this problem as follows. There are K commodities, each with given demands to be satisfied. There are also M coupling constraints bounding from above weighted sums of edge flows over sets of edges within the same commodity or across several commodities, provided that there are at most S positive entries in each partition of a coupling row corresponding to any block. That is, there are at most S edges that enter in any one such generalized bounding constraint. With no loss of generality each arc in the network is assumed to have capacity λ. It is required to find a flow that minimizes λ. Our algorithm reduces this problem to expected $O(SK \ln M(\varepsilon^{-2} + \ln SK))$

single-commodity minimum-cost flow computations. The case $S = 1$ includes the concurrent-flow problem considered in [16] and [7], but still results in a better bound. If, in addition, the individual commodity blocks are generalized flow (gain-loss) networks, this is also the expected number of minimum-cost generalized network flow problems to be solved.

Generalized supply sharing transportation network. Consider the following resource sharing problem for a capacitated generalized (gain-loss) bipartite transportation network with m arcs and $n_1, n_2 \geq 2$ sources and sinks, respectively. There are given demands $b_j > 0$ for each sink j, and a source utilization factor $a_i > 0$, for each source i. It is required to meet the demands by shipping the least amount of total supply from the sources in the given proportions a_i. This is a linear program $(\mathcal{P})$ with $S = 1$, $M = n_1$, and $K = n_2$, also known as the *machine loading* problem. By Theorem 3, the expected number of block problems to be solved is $O(n_2 \ln n_1(\epsilon^{-2} + \ln n_2))$. Each block problem is a capacitated minimum-cost star-tree problem of one sink j and d_j sources, where d_j is the degree of sink j. This block problem can be trivially solved in $O(d_j \ln d_j)$ sequential time, but it can also be solved in $O(d_j)$ time (Brucker [1]). Averaging this time over the sinks for the randomized variant of our algorithm, we obtain an amortized time of $O(d)$, where d is the average degree of a sink in the network. This sharing problem can thus be solved in expected $O(m \ln n_1(\epsilon^{-2} + \ln n_2))$ sequential time. If the block problems are solved on n_1 parallel processors, the resulting expected time is $O(\varepsilon^{-2} n_2)$ up to logarithmic factors in n_1 and n_2. This problem is a generalization for the bipartite vcase of the minimax maximum-flow sharing problem described by Gallo, Grigoriadis, and Tarjan [6].

Generalized quadratic transportation problems. The feasibility form of such problems is similar to the previous model. It calls for computing a flow that satisfies given supplies and demands to a given relative accuracy, and such that a given convex-separable quadratic function of the arc flows (possibly including linear terms) does not exceed a given threshold. This is a nonlinear problem $(\mathcal{P})$ with $S = n_1 \geq 2$, the number of sources, $K = n_2 \geq 2$, the number of sinks, and $M = n_1 + 1$. Each block problem is now a capacitated quadratic minimum-cost star-tree problem, which again can be solved in $O(d_j)$ time using a parametric method with median search [1]. Thus we obtain the same bounds as those in the previous paragraph multiplied by a factor of n_1. In particular, the problem can be solved on n_1 processors in $O(\varepsilon^{-2} n_1 n_2)$ expected time, up to logarithmic factors in n_1 and n_2.

Objective and constraint functions of arbitrary sign. Suppose that the objective and constraint functions of the nonlinear optimization problem (1.2) of §1.1 are not necessarily nonnegative, but that an a priori bound $|f_m^k(x^k)| \leq V_m^k$, $x^k \in B^k$ is known for each block k and each $m = 0, 1, \ldots, M$, so that $|f_m(x)| \leq V_m = \sum_{k=1}^{K} V_m^k$ for all $x \in B$. Define an *ε-approximate solution* to (1.2) to be an arbitrary point $x \in B$ that satisfies $f_0(x) \leq f^* + V_0\varepsilon$, $f_m(x) \leq b_m + V_m\varepsilon$ for all m. By replacing $f_m^k(x^k)$ by $f_m^k(x^k) + 2V_m^k$, the problem of computing such a solution can be reduced to the original problem $(\mathcal{P}_{\varepsilon/3})$. Furthermore, letting $q = 3$ effectively removes the constraints $f_m^k(x^k) + 2V_m^k \leq \mu q e$ in the definition (3.1) of $B_{\mu q}^k$ since $\mu \geq V_m \geq V_m^k$.

Thus, the $(\varepsilon', 3)$-good block solver can operate on the original sets B^k to minimize $p^T f^k(x^k)$ to an *absolute* accuracy $\delta = \varepsilon' p^T V^k$ or $\delta = O(\varepsilon p^T V^k)$ for $\varepsilon' = O(\varepsilon)$ as we have previously done for the nonnegative case. Also, in this case it suffices to call the block solver to find an arbitrary point $x_0 \in B$ instead of using the greedy initialization procedure $\mathcal{GP}$ of §8. Since for the redefined positive functions $\lambda(x_0)/\lambda^* \leq 3$, the bound (7.5) on the number of coordination steps needed to solve $(\mathcal{P}_{\varepsilon/3})$ becomes $O(\varepsilon^{-2} K \ln M)$. For the nonlinear optimization problem (1.2) we obtain the bound

$O(\varepsilon^{-2}K \ln M \ln \varepsilon^{-1})$. As before, each such step generates K block-solver calls for the original sets B^k in the deterministic method, but only two block-solver calls on the average for the randomized variant.

Deterministic equivalents of stochastic programs. The above reduction expands the range of applications of our analysis to many instances of the block-angular structure. Here, we mention the approximate solution of linear and nonlinear deterministic equivalents of two-stage large-scale stochastic programs with discrete finite distributions of the problem data (see Wets [26]; Nielsen and Zenios [17] have recently reported extensive computational results for a mortgage-backed securities investment problem of this type, where uncertainties are modeled by a large set of K "scenarios" based on forecasted interest-rate paths). In general, such problems can be stated as follows:

$$F^* = \min \left\{ f(x) + \sum_{k=1}^{K} \pi^k g^k(y^k) \,|x \in A,\ (x, y^k) \in S^k \right\},$$

where x is the first-stage decision vector, A is a convex compact set in $\mathbb{R}^n$, y^k is the second-stage decision vector corresponding to the kth scenario, and each scenario is given by its postulated probability π^k and a convex compact set S^k as its feasibility domain. The convex functions $f(x)$ and $g^k(y^k)$ are given first- and second-stage objectives. The overall objective is the sum of the deterministic first-stage term plus the expectation of the stochastic objective terms (e.g., quadratic, entropy) over K scenarios. We assume that $A \subseteq [0,1]^n$, possibly after scaling x. We also assume that we are given an a priori bound V^k on the range of possible values of $f(x) + g^k(y^k)$ over $\{(x, y^k) \,|x \in A,\ (x, y^k) \in S^k\}$. Thus, V^k is an a priori bound on the possible "loss" in the kth scenario. Let $V = \sum_{k=1}^{K} \pi^k V^k$. We shall say that $z = (x, y^1, \ldots, y^k)$ is an *ε-approximate solution* if $f(x) + \sum_{k=1}^{K} \pi^k g^k(y^k) \leq F^* + V\varepsilon$, $x \in A$, and each pair (x, y^k) is *ε-feasible* for the kth scenario in the sense that there is a perturbation vector δ^k such that $\|\delta^k\|_\infty \leq \varepsilon$ and $(x + \delta^k, y^k) \in S^k$. In practical terms, the latter condition corresponds to requiring that the postulated scenarios are sufficiently robust with respect to the first-stage decision. Since our problem can be rewritten as

$$F^* = \min \left\{ \sum_{k=1}^{K} \pi^k (f(x^k) + g^k(y^k)) \,|x^1 = x^k, \text{ and } (x^k, y^k) \in B^k,\ k = 1, \ldots, K \right\},$$

where $B^k = \{(x^k, y^k) \mid x^k \in A,\ (x^k, y^k) \in S^k\}$, it can be considered as a block-angular problem with blocks B^k and with $2(K-1)n$ linear coupling functions $\pm(x_j^1 - x_j^k)$.

The algorithm of the previous application computes K pairs $(x^k, y^k) \in B^k$ such that $\sum_{k=1}^{K} \pi^k (f(x^k) + g^k(y^k)) \leq F^* + V\varepsilon$ and $\|x^1 - x^k\|_\infty \leq \varepsilon$, $k = 1, \ldots, K$, in expected $O(\varepsilon^{-2}K \ln Kn \ln \varepsilon^{-1})$ iterations. At each iteration, the algorithm minimizes the sum of $f(x^k) + g(y^k)$ plus a linear form in x^k, over $(x^k, y^k) \in B^k$ for a randomly selected scenario k. Since f is convex, $z = (\sum_{k=1}^{K} \pi^k x^k, y^1, \ldots, y^K)$ provides an ε-approximate solution to our original problem.

Fast parallel approximation algorithm for matrix games. Let A be an $N \times M$ matrix game whose entries belong to an interval $[\alpha, \beta]$ and let v^* be the value of the game. Consider the problem $VAL(\varepsilon)$ of computing an approximation v to v^* to a prescribed relative accuracy $\varepsilon \in (0,1)$, that is, $|v - v^*| \leq \varepsilon(\beta - \alpha)$, and of computing the corresponding pair of mixed ε-optimal strategies.

Assume without loss of generality that $[\alpha, \beta] = [1, 2]$. Applying Theorem 3 for the case $K = 1$, $B = P$, and $q = 2$, along with some additional bounds on the required accuracy of arithmetic operations shown in [9], one can prove the following result.

THEOREM 4. *There is a parallel algorithm for VAL(ε) whose execution uses NM processors and has depth of $O(\varepsilon^{-2}\ln^2 NM)$ arithmetic operations on $O(\ln(NM\varepsilon^{-1}))$-bit numbers.*

Acknowledgments. We thank Gennady Samorodnitsky for an earlier proof of Lemma 3 mentioned in §6, and Éva Tardos for providing us with a preliminary version of [16].

Note added in proof. The method presented in this paper has been extended to the general block-angular optimization problem (1.2) in a way that bisection search on the objective function value is avoided. This is accomplished by reducing problem (1.2) to two resource-sharing problems of the form (1.1). This reduction, as well as some computational results are reported in [10].

REFERENCES

[1] P. BRUCKER, *An $O(n)$ algorithm for quadratic knapsack problems*, Oper. Res. Lett., 3 (1984), pp. 163–166.

[2] G. B. DANTZIG AND P. WOLFE, *The decomposition algorithm for linear programming*, Econometrica, 29 (1961), pp. 767–778.

[3] H. EVERETT III, *Generalized Lagrange multiplier method for solving problems of optimum allocation of resources*, Oper. Res., 11 (1963), pp. 399–417.

[4] M. FRANK AND P. WOLFE, *An algorithm for quadratic programming*, Naval Res. Logistics Quarterly, 3 (1956), pp. 149–154.

[5] L. FRATTA, M. GERLA, AND L. KLEINROCK, *The flow-deviation method: an approach to store-and-forward communication network design*, Networks, 3 (1973), pp. 97–133.

[6] G. GALLO, M. D. GRIGORIADIS, AND R. E. TARJAN, *A fast parametric maximum flow algorithm and applications*, SIAM J. Comput., 18 (1989), pp. 30–55.

[7] A. GOLDBERG, *A natural randomization strategy for multicommodity flow and related problems*, manuscript, August 1991.

[8] M. D. GRIGORIADIS AND L. G. KHACHIYAN, *Fast approximation schemes for convex programs with many blocks and coupling constraints*, Tech. Rep. 91-24, DIMACS (also DCS-TR-273, Dept. of Computer Science), Rutgers Univ., New Brunswick, NJ, March 1991.

[9] ———, *Approximate solution of matrix games in parallel*, in Advances in Optimization and Parallel Computing – Honorary Volume on the Occasion of J. B. Rosen's 70th Birthday, P. M. Pardalos, ed., Elsevier, 1992.

[10] ———, *An exponential-function reduction method for block-angular convex programs*, Tech. Rep. LCSR-TR-211, Dept. of Computer Science, Rutgers Univ., New Brunswick, NJ, May 1993.

[11] M. D. GRIGORIADIS AND W. W. WHITE, *A partitioning algorithm for the multicommodity network flow problem*, Math. Programming, 3 (1972), pp. 157–177.

[12] R. M. KARP, *Probabilistic recurrence relations*, in Proc. 23rd Annual ACM Symp. on Theory of Computing, pp. 190–197, May 1991.

[13] L. G. KHACHIYAN, *Convergence rate of the game processes for solving matrix games*, Zh. Vȳchisl. Mat. & Mat. Fiz., 17-6 (1977), pp. 1421–1431. English translation in U.S.S.R. Comput. Math. & Math. Phys., 17 (1978), pp. 78–88.

[14] P. KLEIN, C. STEIN, AND E. TARDOS, *Leighton–Rao might be practical: faster approximation algorithms for concurrent flow with uniform capacities*, in Proc. 22nd Annual ACM Symp. on Theory of Computing, pp. 310–321, May 1990.

[15] L. S. LASDON, *Optimization Theory for Large Systems*, The MacMillan Company, New York, NY, 1970.

[16] T. LEIGHTON, F. MAKEDON, S. PLOTKIN, C. STEIN, E. TARDOS, AND S. TRAGOUDAS, *Fast approximation algorithms for multicommodity flow problems*, in Proc. 23rd Annual ACM Symp. on Theory of Computing, pp. 101–111, May 1991.

[17] S. S. NIELSEN AND S. A. ZENIOS, *A massively parallel algorithm for nonlinear stochastic network problems*, Report 90-09-08, The Wharton School, Univ. of Pennsylvania, Philadelphia, PA , October 1990.

[18] S. A. PLOTKIN, D. B. SHMOYS, AND E. TARDOS, *Fast approximation algorithms for fractional packing and covering problems*, in Proc. 32nd Annual Symp. on Foundations of Computer Science, pp. 495–504, October 1991.

[19] J. B. ROSEN, *Primal partition programming for block diagonal matrices*, Numer. Math., 6 (1964), pp. 250–260.

[20] J. ROBINSON, *An iterative method of solving a game*, Ann. of Math., 54 (1951), pp. 196–301.

[21] F. SHAHROKHI AND D. W. MATULA, *The maximum concurrent flow problem*, J. ACM, 37 (1990), pp. 318–334.

[22] T. R. ROCKAFELLAR, *Convex Analysis*, Princeton University Press, Princeton, NJ, 1970.

[23] G. P. SAMORODNITSKY, Personal communication (December 1990).

[24] G. L. SCHULTZ AND R. R. MEYER, *An interior point method for block-angular optimization*, SIAM J. Optimization, 1 (1991), pp. 583–602.

[25] E. TARDOS, *Improved approximation algorithm for concurrent multicommodity flows*, Tech. Rep. 872, School of OR/IE, Cornell Univ., Ithaca, NY, October 1989.

[26] R. WETS, *Numerical Techniques for Stochastic Optimization*, Springer-Verlag, Berlin, 1988.

SIAM J. OPTIMIZATION
Vol. 4, No. 1, pp. 108–129, February 1994
© 1994 Society for Industrial and Applied Mathematics
005

ON THE USE OF PRODUCT STRUCTURE IN SECANT METHODS FOR NONLINEAR LEAST SQUARES PROBLEMS*

J. HUSCHENS†

Abstract. The problem of adjusting the second-order term in secant methods for nonlinear least squares problems with zero-residual is addressed. The author uses the framework of structured secant methods to derive and investigate a new way to resize the approximation of the second-order term using some exact information. The resulting method is a self-adjusting structured secant method for nonlinear least squares problems, yielding a q-quadratic convergence rate for zero residual and a q-superlinear convergence rate for nonzero residual problems.

Key words. nonlinear least squares, structured secant method, convergence theory, quadratic convergence, bounded deterioration

AMS subject classifications. 65K, 49D

1. Introduction.

In this paper we consider the nonlinear least squares problem

$$\text{minimize} \quad f(x) = \frac{1}{2}R(x)^T R(x) = \frac{1}{2}\sum_{i=1}^{m} r_i(x)^2, \tag{1.1}$$

where the residual function $R : \mathbb{R}^n \to \mathbb{R}^m$ $(m \geq n)$ is nonlinear and $r_i(x)$ denotes the ith component of $R(x)$. Many algorithms for solving problem (1.1) have been proposed, most of them somehow taking the structure of the problem into account (see, e.g., [3], [5], [11], and [14]). Since

$$\nabla f(x) = J(x)^T R(x), \tag{1.2}$$

where $J(x)$ denotes the Jacobian of R at x, the Hessian of f is given by

$$\nabla^2 f(x) = C(x) + S(x) \tag{1.3}$$

with

$$C(x) = J(x)^T J(x) \quad \text{and} \quad S(x) = \sum_{i=1}^{m} r_i(x)\, \nabla^2 r_i(x). \tag{1.4}$$

This inherent structure leads to quadratic convergence of the Gauss–Newton method in the zero-residual case, but $S(x)$ gives problems to first-order methods in the case of large residuals or strong nonlinearities. To overcome these difficulties and still maintain the quadratic convergence rate for zero-residual problems, hybrid methods for the nonlinear least squares problem have been proposed [1], [15]. The critical point in these hybrid methods is the criterion controlling the switching between two different algorithms in order to obtain the desired convergence properties.

Another approach to getting benefits out of the underlying structure is given by the theory of structured secant methods [6], [7], [11], [12], [13]. Dennis, Martínez, and Tapia [7] stated a "structure principle" for using given additive structure in a secant algorithm. The application of this structure principle to problem (1.1) leads

* Received by the editors April 25, 1991; accepted for publication (in revised form) November 18, 1992.

† Universität Trier, Fachbereich IV–Mathematik, Postfach 3825, 54296 Trier, Germany.

to the structured secant method that is currently used in the NL2SOL-code with a BFGS-update. This structured secant method for the nonlinear least squares problem uses the first-order information of $C(x)$ in (1.3) exactly, while the second-order term $S(x)$ is approximated using the framework of secant methods (for a genealogy of this method see [19]). Using only the additive structure in the form of (1.3) this method yields local q-superlinear convergence in x, but gives no q-quadratic convergence in the case of zero-residual problems. To address this problem in the NL2SOL-code an Oren–Luenberger scaling technique is used [6], [11], [21], but there is no theoretical background available for this way of adjusting the second-order approximation in the zero-residual case.

In order to avoid these problems we need an approximation of $\nabla^2 f(x)$ using only exact first-order information, that will yield q-superlinear convergence for nonzero-residual problems, but also q-quadratic convergence in the zero-residual case. To find the desired approximation we rewrite $\nabla^2 f(x)$ in a simplistic way that makes self-scaling possible by incorporating exact parts:

$$\nabla^2 f(x) = C(x) + \|R(x)\| \sum_{i=1}^{m} \frac{r_i(x)}{\|R(x)\|} \nabla^2 r_i(x). \tag{1.5}$$

So the explicit use of the first-order part $C(x)$ and also of $\|R(x)\|$ results in the following structured approximation of $\nabla^2 f(x)$:

$$B = C(x) + \|R(x)\| A\,, \tag{1.6}$$

where only the part

$$T(x) = \sum_{i=1}^{m} \frac{r_i(x)}{\|R(x)\|} \nabla^2 r_i(x) \tag{1.7}$$

remains to be approximated by a suitable matrix A.

This task implies some difficulties, since we have to balance the approximation with respect to either the nonzero- or the zero-residual case. An indication for the arising problems is the fact that a priori we cannot define $T(x_*)$ for a local solution x_* with $\|R(x_*)\| = 0$.

So the simple idea of (1.5) that carries over to other applications (e.g., in constrained optimization) corresponds to strong requirements on the choice of the approximation matrices for $T(x)$.

The purpose of this paper is to demonstrate how a product-structure technique based on the ideas in (1.5) and (1.6) can be exploited by a structured secant method using additive and multiplicative structure. The exact use of the multiplicative structure results in local q-quadratic convergence in the zero-residual case, but the proposed way of choosing the approximation matrices also leads to local q-superlinear convergence in the nonzero-residual case. The resulting algorithm, which will be formulated in §2, therefore possesses the best convergence properties that can be achieved if no exact second-order information is used. Furthermore, unlike the hybrid methods proposed for the nonlinear least squares problem [1], [15], in this algorithm there is no need to switch between two different algorithms. In §3 we will prove the local q-quadratic convergence in the zero-residual case. This proof offers some new features with respect to the convergence theory of secant methods, since merely calculations about the scale (for the notion of "scale" in this context, see (1.8) below) will lead to

a bounded deterioration property in a special weighted Frobenius norm corresponding to the update used. In the §4 we will prove the q-superlinear convergence of the proposed algorithm in the nonzero-residual case using the convergence theory of structured secant methods as developed in [7], [13], and [18] and a new convenient derivation of the Dennis–Moré characterization of q-superlinear convergence. Computational comparisons illustrating the numerical behavior of the method will be given in §5. But first we will make some technical remarks concerning the updates used in the algorithm and state some standard assumptions.

Throughout this paper $\|\cdot\|$ denotes the ℓ_2-norm or the induced operator norm. In the convergence theory we will consider the weighted Frobenius norm $\|\cdot\|_{F,M} = \|M \cdot M\|_F$ for a symmetric, positive-definite matrix $M \in \mathbb{R}^{n\times n}$. If we use $B_*^{-1/2} = \nabla^2 f(x_*)^{-1/2}$ as the weighting matrix, the corresponding weighted Frobenius norm will be denoted by $\|\cdot\|_*$.

In our context we will use the least change formulation of secant updates [9], [10]

$$\triangle_2(s,y,B,v) = \frac{(y-Bs)v^T + v(y-Bs)^T}{v^Ts} - \frac{(y-Bs)^Tsvv^T}{(v^Ts)^2}, \tag{1.8}$$

because the scale v offers the opportunity to transfer information on the whole problem setting to the update of a structured part (compare to the structure principle in [7]). Therefore we will choose the scale depending on the relevant information, i.e., $v = v(s,y,B)$. The $\triangle_2$ -update is invariant to the size of the scale, that is

$$\triangle_2(s,y,B,\alpha v) = \triangle_2(s,y,B,v) \quad \forall \alpha \in \mathbb{R}\setminus\{0\}. \tag{1.9}$$

Therefore a rule for resizing the approximated part seems to be reasonable.

In the following lemma we will state a relationship between the least change secant updates for special choices of the scale and the updates from the convex Broyden class:

$$\triangle_1(s,y,B,\phi) = \frac{yy^T}{y^Ts} - \frac{Bss^TB}{s^TBs} + \phi s^TBsuu^T \tag{1.10}$$

with

$$u = \frac{y}{y^Ts} - \frac{Bs}{s^TBs} \quad \text{and} \quad \phi \in [0,1].$$

LEMMA 1.1. *Let $y, s \in \mathbb{R}^n$ with $y^Ts > 0$; $B \in \mathbb{R}^{n\times n}$ symmetric with $s^TBs > 0$. Then for each*

$$\sigma \in \left[0, \left[\frac{y^Ts}{s^TBs}\right]^{\frac{1}{2}}\right]$$

the scale $v = y + \sigma Bs$ in (1.8) *yields an update from the convex Broyden class, and the function Φ:*

$$\Phi(\sigma) = \frac{(y^Ts)^2 - \sigma^2 y^Tss^TBs}{(s^T(y+\sigma Bs))^2} \tag{1.11}$$

is bijective from

$$\left[0, \left[\frac{y^Ts}{s^TBs}\right]^{\frac{1}{2}}\right] \quad \text{to } [0,1].$$

Furthermore, it holds that

$$\triangle_2(s, y, B, y + \sigma Bs) = \triangle_1(s, y, B, \Phi(\sigma)). \tag{1.12}$$

For a proof see [18], but notice corresponding results in [11] and [22].

Therefore we can reparametrize the convex Broyden class by using the least change formulation (1.8) with the scales $v = y + \tau\rho Bs$, where

$$\tau \in [0,1] \quad \text{and} \quad \rho = \left[\frac{y^T s}{s^T B s}\right]^{\frac{1}{2}}.$$

For $\tau = 1$ we get the well-known BFGS-update, while $\tau = 0$ yields the DFP-update.

The following standard assumptions for the nonlinear least squares problem (1.1) shall hold for the rest of this paper.

(A1) The nonlinear least squares problem (1.1) has a local solution x_*, i.e., there exists a constant $\varepsilon_1 > 0$ such that $\|R(x_*)\| \leq \|R(x)\|$ for all $x \in D_1 = \{x : \|x - x_*\| < \varepsilon_1\}$.

(A2) Let the function f be in C^2 and let $\nabla^2 f(x), C(x), J(x)$, and $R(x)$ be Lipschitz-continuous in the neighborhood D_1 of x_*, i.e., there exist nonnegative constants L, L_C, L_J, L_R, such that for all $x, x_+ \in D_1$:

$$\begin{aligned} \|\nabla^2 f(x) - \nabla^2 f(x_*)\| &\leq L\,\|x - x_+\|, \\ \|C(x) - C(x_*)\| &\leq L_C \|x - x_+\|, \\ \|J(x) - J(x_*)\| &\leq L_J \|x - x_+\|, \\ \|R(x) - R(x_*)\| &\leq L_R \|x - x_+\|. \end{aligned}$$

(A3) The matrix $\nabla^2 f(x_*)$ is positive definite, i.e., there exist positive constants m, M :

$$m\|z\|^2 \leq z^T \nabla^2 f(x_*) z \leq M\|z\|^2 \quad \forall\, z \in \mathbb{R}^n.$$

These assumptions ensure the second-order sufficiency condition for x_*. For convenience we have stated conditions on the Lipschitz-continuity of $C(x)$ in (A2), although these are already implied by those for $J(x)$.

2. Derivation of the method. Since it is our strategy to approximate

$$\nabla^2 f(x) = C(x) + \|R(x)\| \sum_{i=1}^{m} \frac{r_i(x)}{\|R(x)\|} \nabla^2 r_i(x) \tag{2.1}$$

with matrices of the form

$$B = C(x) + \|R(x)\| A\,, \tag{2.2}$$

the well-developed theory of structured secant methods offers an appropriate way of handling and updating these matrices A and B. Furthermore, the theory of structured secant methods makes it possible to maintain the secant equation and to obtain a satisfying local behavior for this kind of method.

The use of the exact part $\|R(x)\|$ yields a self-scaling property that will, at the very least locally, give the right tendency for sizing the approximation to the part $\sum_{i=1}^{m} r_i(x) \nabla^2 r_i(x)$. In the framework of secant methods for a step s we have to find a good approximation $y^{\#}$ to $T(x_+)s$ (see (1.7)), because all secant methods depend

heavily on the quality of these approximations. Since $(J(x_+) - J(x))^T R(x_+)$ is a good approximation to $\|R(x_+)\|T(x_+)s$ (see [2], [6], and [7]), in our context we choose $y^\# = (J(x_+) - J(x))^T(R(x_+)/\|R(x)\|)$ as an approximation to $T(x_+)s$. This choice reflects the relation between $T(x)s$ and $T(x_+)s$, since $y^\#$ should also be a reasonable approximation to $T(x)s$. At this point it is obvious to ask why we are dividing $R(x_+)$ by $\|R(x)\|$ rather than by $\|R(x_+)\|$. In §§3 and 4 it will be shown that the given choice is appropriate. The alternative of replacing $\|R(x)\|$ by $\|R(x_+)\|$ in the denominator of $y^\#$ does not allow us to prove q-quadratic convergence in the zero-residual case, since $y^\#$ with this definition does not fulfill the required approximation quality. It should be noticed that $y^\#$ is well defined and, with some care, well computable, since we can stop the algorithm if $\|R(x_+)\|$ is smaller than a number near the machine precision.

Applying the structure principle developed in [7] to the problem formulation (2.1) with approximations in the form of (2.2) and choosing $y^\#$ as above, we propose the following.

Totally structured secant method for unconstrained nonlinear least squares problems (TSSM). Given $x \in \mathbb{R}^n, A \in \mathbb{R}^{n\times n}$ symmetric such that $B = J(x)^T J(x) + \|R(x)\|A \in \mathbb{R}^{n\times n}$, solve

$$Bs = -\nabla f(x) \tag{2.3}$$

and set

$$x_+ = x + s, \tag{2.4}$$

$$y^\# = (J(x_+) - J(x))^T \frac{R(x_+)}{\|R(x)\|}, \tag{2.5}$$

$$y = J(x_+)^T J(x_+)s + \|R(x_+)\|y^\#, \tag{2.6}$$

$$B^s = J(x_+)^T J(x_+) + \|R(x_+)\|A, \tag{2.7}$$

$$A_+ = A + \triangle_2(s, y^\#, A, v(s, y, B^s)), \tag{2.8}$$

$$B_+ = J(x_+)^T J(x_+) + \|R(x_+)\|A_+. \tag{2.9}$$

In the sequel we need a special property of this algorithm. If it is defined that $\hat{y} := (\|R(x_+)\|/\ \|R(x)\|)(J(x_+) - J(x))^T R(x_+)$ we obtain, with (2.7)–(2.9),

$$y - B^s s = \|R(x_+)\|y^\# - \|R(x_+)\|As = \hat{y} - \|R(x_+)\|As.$$

With the definition of the $\triangle_2$-update in (1.8) this yields

$$\|R(x_+)\| \triangle_2 (s, y^\#, A, v) = \triangle_2(s, \hat{y}, \|R(x_+)\|A, v) = \triangle_2(s, y, B^s, v).$$

Therefore we obtain

$$B_+ = B^s + \triangle_2(s, y, B^s, v)\,. \tag{2.10}$$

Next we want to show that the method given above possesses desirable convergence properties.

3. Q-quadratic convergence for zero-residual problems. We will split the convergence theory of the proposed method (2.3)–(2.9) into two parts: the convergence theory for zero-residual problems and the convergence theory for nonzero-residual problems. These cases are treated separately, since their analysis is based on different techniques. While the proof of the local q-quadratic convergence in the zero-residual

case is based on an inductive argument and gives a derivation of the bounded deterioration property merely by calculations on the scale, the proof of the local q-superlinear convergence in the nonzero-residual case emanates in a more standard way from the general convergence theory of structured secant methods. Therefore we will start with the zero-residual case. In the following we will use the notation $\sigma(x, x_+) = \max\{\|x - x_*\|, \|x_+ - x_*\|\}$ and assume for convenience that finite convergence does not occur, i.e., $x_k \neq x_*$ for all $k \in \mathbb{N}_0$.

THEOREM 3.1. *Suppose that the standard assumptions* (A1)–(A3) *hold in a zero-residual case, i.e.,* $\|R(x_*)\| = 0$. *Then there exist constants* $\varepsilon, \delta > 0$, *such that for* $\|x_0 - x_*\| < \varepsilon$ *and* $A_0 \in \mathbb{R}^{n\times n}$ *symmetric with* $\|A_0\| < \delta$ *the sequence of iterates* x_k *generated by the totally structured secant method* (TSSM) (2.3)–(2.9) *using a scale of the form*

$$v_k = y_k + \tau\rho_k B_k^s s_k \quad \textit{with fixed} \quad \tau \in [0,1] \quad \textit{and} \quad \rho_k = \left(\frac{y_k^T s_k}{s_k^T B_k^s s_k}\right)^{\frac{1}{2}}$$

is well defined and converges at a q-quadratic rate to x_*.

Proof. Let $B_* = \nabla^2 f(x_*) = J(x_*)^T J(x_*)$. Choose ε, δ sufficiently small, such that with the constants given in (A1)–(A3)

$$\delta \leq \frac{m(1+\tau)}{\sqrt{n}}, \qquad \varepsilon \leq \varepsilon_* \tag{3.1}$$

holds for a positive constant ε_* satisfying the condition

(3.2)
$$\varepsilon_* < \min\left\{\frac{m}{L_C + L_R(1+\tau)M}, \frac{m - \varepsilon(L_C + L_R(1+\tau)M)}{K}, \frac{m}{L}, \frac{m}{K_y}, \frac{1}{L_R K^*}, \frac{1}{3K_v}, \varepsilon_1\right\},$$

where with the definitions

$$\begin{aligned} \gamma_* &:= \varepsilon_*(L_C + L_R(1+\tau)M), \qquad \Theta_* := \frac{1}{m - \gamma_*}, \\ \gamma &:= \varepsilon(L_C + L_R(1+\tau)M), \qquad \Theta := \frac{1}{m - \gamma} \end{aligned} \tag{3.3}$$

(all these quantities are positive by the restrictions above), the constants K, K_y, K_v, K^* and $K_{v^T s}$, arising later in the proof, are given by

(3.4)
$$K := \frac{L_J^2}{2}\varepsilon_* + \|J(x_*)\|\frac{L_J}{2} + L_R(1+\tau)M, \quad K_y := L_C + \frac{L_R^2}{\sqrt{\alpha}}L_J, \quad \alpha := \frac{1}{2}(m - L\varepsilon_*),$$

$$K_v := \frac{K_y}{m(1+\tau)} + \tau\sqrt{\frac{M + K_y\varepsilon_*}{m - \gamma_*}}\frac{L_C + L_R(1+\tau)M}{m(1+\tau)} + \frac{\tau}{1+\tau}\sqrt{\frac{M}{m}}L_R K^*,$$

$$K^* := \frac{L_J\frac{L_R}{\sqrt{\alpha}} + M(1+\tau)}{m - \gamma_*}, \quad K_{v^T s} := (m - K_y\varepsilon_*) + \tau\sqrt{\frac{m - K_y\varepsilon_*}{M + \gamma_*}}(m - \gamma_*),$$

$$K_{\|v\|} := K_y\varepsilon_* + M + \sqrt{\frac{M + K_y\varepsilon_*}{m - \gamma_*}}(M + \gamma_*).$$

Note that (3.2) can be satisfied, since γ_* in (3.3) gets small as ε_* gets small. Furthermore, since by (3.1) and (3.2)

$$\varepsilon < \frac{m}{K + L_C + L_M(1+\tau)M},$$

we obtain, with the definition $\bar{\Theta} := \Theta K \varepsilon$ and by (3.2),

$$\bar{\Theta} < 1. \tag{3.5}$$

Finally, if necessary, reduce ε and δ so that the combined condition on ε and δ

$$e^{\max\{\alpha_1,\alpha_2,\alpha_3\}\varepsilon/(1-\bar{\Theta})}\left(\delta\frac{\sqrt{n}}{m(1+\tau)} + \max\{\alpha_1,\alpha_2,\alpha_3\}\varepsilon\frac{1}{1-\bar{\Theta}}\right) < 1 \tag{3.6}$$

is satisfied, where α_1, α_2, and α_3, are defined by

$$\alpha_1 = \frac{5}{2}\frac{K_v}{1 - K_v\varepsilon_*}, \qquad \alpha_2 = \frac{2(1+2\sqrt{n})}{\sqrt{m}}\|B_*^{-\frac{1}{2}}\|_F L_J \frac{L_R}{\sqrt{\alpha}}\Theta_* K,$$
$$\alpha_3 = \left(2 + \frac{K_{\|v\|}}{K_v \tau_s}\right)\frac{L_J L_R \Theta_* K_{\|v\|} K}{(1+\tau)mK_v\tau_s\sqrt{\alpha}}.$$

In the following analysis we will use the weighting matrix

$$M_\tau = \frac{1}{\sqrt{1+\tau}}B_*^{-\frac{1}{2}}. \tag{3.7}$$

Since the weighted Frobenius norm induced by the matrix M_τ is equivalent to the ℓ_2-operator norm with

$$\frac{m(1+\tau)}{\sqrt{n}}\|A\|_{F,M_\tau} \le \|A\| \le (1+\tau)M\|A\|_{F,M_\tau}, \tag{3.8}$$

the assumption (3.1) on δ implies

$$\|A_0\|_{F,M_\tau} < 1 \quad \text{and therefore by (3.8),} \quad \|A_0\| \le (1+\tau)M. \tag{3.9}$$

Using (3.9) we conclude with assumption (A2) and the definitions in (3.3) that

$$\begin{aligned}\|B_0 - B_*\| &\le L_C\|x_0 - x_*\| + L_R\|x_0 - x_*\|\|A_0\| \\ &\le (L_C + L_R(1+\tau)M)\|x_0 - x_*\| < \gamma.\end{aligned} \tag{3.10}$$

Therefore B_0 is positive definite, since we have by (A3), (3.2), (3.10) and $\gamma \le \gamma_*$,

$$z^T B_0 z \ge m\|z\|^2 - |z^T(B_0 - B_*)z| \ge (m - \gamma_*)\|z\|^2.$$

Recall that by the Banach perturbation lemma (e.g., in [11, Thm. 3.1.4]) we have

$$\|B^{-1}\| \le \Theta \qquad \forall\, B \in \mathbb{R}^{n\times n} \quad \text{with} \quad \|B - B_*\| < \gamma. \tag{3.11}$$

With the restrictions on ε and δ given in (3.1), (3.2), (3.5), (3.6) and the notation in (3.3), we will prove the following by induction for all $i \in \mathbb{N}_0$:

$$\begin{aligned}&(1) \quad \|x_{i+1} - x_*\| \le \Theta K\|x_i - x_*\|^2 \le \bar{\Theta}\|x_i - x_*\|, \\ &(2) \quad \|A_{i+1}\|_{F,M_\tau} \le (1 + \alpha_1\sigma(x_i, x_{i+1}))\|A_i\|_{F,M_\tau} + \alpha_2\sigma(x_i, x_{i+1}), \\ &(3) \quad \|A_{i+1}\|_{F,M_\tau} < 1, \\ &(4) \quad \|B_{i+1} - B_*\| < \gamma.\end{aligned} \tag{3.12}$$

(1) To show the quadratic reduction we conclude that

$$\begin{aligned} x_1 - x_* &= x_0 - x_* - (J(x_0)^T J(x_0) + \|R(x_0)\|A_0)^{-1} J(x_0)^T R(x_0) \\ &= -(J(x_0)^T J(x_0) + \|R(x_0)\|A_0)^{-1}\{J(x_0)^T R(x_0) \\ &\quad -(J(x_0)^T J(x_0) + \|R(x_0)\|A_0)(x_0 - x_*)\} \\ &= -(J(x_0)^T J(x_0) + \|R(x_0)\|A_0)^{-1}\{J(x_0)^T R(x_*) \\ &\quad -J(x_0)^T[R(x_*) - R(x_0) - J(x_0)(x_* - x_0)] - \|R(x_0) - R(x_*)\|A_0(x_0 - x_*)\}. \end{aligned}$$

Taking the norms we conclude, using an application of the mean value theorem in the form of Lemma 4.1.12 in [11], the estimates in (A2), (3.3), (3.9), (3.11) and $\|R(x_*)\| = 0$:

(3.13)

$$\begin{aligned} \|x_1 - x_*\| &\le \|(J(x_0)^T J(x_0) + \|R(x_0)\|A_0)^{-1}\| \left(\|J(x_0)\|\frac{L_J}{2} + L_R\|A_0\|\right)\|x_0 - x_*\|^2 \\ &\le \Theta\left(\frac{L_J^2}{2}\varepsilon + \|J(x_*)\|\frac{L_J}{2} + L_R(1+\tau)M\right)\|x_0 - x_*\|^2 = \Theta K\|x_0 - x_*\|^2 \\ &\le \bar{\Theta}\|x_0 - x_*\|. \end{aligned}$$

(2) In order to show the asserted bounded deterioration property of the matrices A_k we need some further investigation.

(a) By the mean value theorem we get, for a $\vartheta(x) \in [0,1]$,

$$\|R(x)\|^2 = \tfrac{1}{2}(x - x_*)^T \nabla^2 f(x_* + \vartheta(x)(x - x_*))(x - x_*) \ge \alpha\|x - x_*\|^2$$

with $\alpha = \frac{1}{2}(m - L\varepsilon_*) > 0$, because by assumptions (A2), (A3), and (3.2) we get

$$\begin{aligned} z^T \nabla^2 f(x) z &= z^T \nabla^2 f(x_*) z + z^T(\nabla^2 f(x) - \nabla^2 f(x_*))z \\ &\ge (m - L\varepsilon_*)\|z\|^2 \quad \text{for } \|x - x_*\| < \varepsilon. \end{aligned}$$

(b) For $y_0^{\#}$ defined in (2.7) assumption (A2) implies, in connection with (3.13) and part (a),

(3.14)

$$\begin{aligned} \|y_0^{\#}\| &= \left\|[J(x_1) - J(x_0)]^T \frac{R(x_1)}{\|R(x_0)\|}\right\| \\ &\le L_J\|s_0\|\frac{\|R(x_1)\|}{\|R(x_0)\|} \le L_J\|s_0\|\frac{L_R\|x_1 - x_*\|}{\sqrt{\alpha}\|x_0 - x_*\|} \le L_J\frac{L_R}{\sqrt{\alpha}}\Theta K\sigma(x_0, x_1)\|s_0\|. \end{aligned}$$

This yields

(3.15)

$$\begin{aligned} \|y_0 - B_* s_0\| &\le \|J(x_1)^T J(x_1) - J(x_*)^T J(x_*)\|\|s_0\| + \|R(x_1)\|\|y_0^{\#}\| \\ &\le K_y\sigma(x_0, x_1)\|s_0\| \quad \text{with } K_y = L_C + \frac{L_R^2}{\sqrt{\alpha}} L_J \end{aligned}$$

for y_0 defined in (2.8), together with (3.5) and (3.13).

(c) Since B_0 is positive definite by (3.10) and we can stop, if $R(x_1) = 0$, we have to ensure $s_0^T B_0^s s_0 > 0$, $y_0^T s_0 \ge 0$ and $v_0^T s_0 \ne 0$ in order to guarantee that the update in (2.6) is well defined. With the same estimates used in (3.10) we can show

$$\|B_0^s - B_*\| \le (L_C + L_R(1+\tau)M)\sigma(x_0, x_1) < \gamma \tag{3.16}$$

for B_0^s defined in (2.9). Therefore, by (A3), (3.2), (3.3), (3.11), and (3.16), B_0^s is positive definite with

$$\|(B_0^s)^{-1}\| \leq \frac{1}{m-\gamma} = \Theta \quad \text{and} \quad z^T B_0^s z \leq (M+\gamma_*)\|z\|^2 \quad \forall z \in \mathbb{R}^n. \tag{3.17}$$

Since by (3.15), $y_0^T s_0 \geq (m - K_y\varepsilon_*)\|s_0\|^2 > 0$, v_0 therefore is well defined, and we get with assumption (A3), (3.2), (3.3), (3.5), (3.15), (3.16), and (3.17),

$$\begin{aligned} v_0^T s_0 &= y_0^T s_0 + \tau\sqrt{\frac{y_0^T s_0}{s_0^T B_0^s s_0}}\, s_0^T B_0^s s_0 \\ &\geq m\|s_0\|^2 - K_y\varepsilon_*\|s_0\|^2 + \tau\sqrt{\frac{m-K_y\varepsilon_*}{M+\gamma_*}}\,(s_0^T(B_0^s - B_*)s_0 + s_0^T B_* s_0) \qquad (3.18) \\ &\geq \left[(m-K_y\varepsilon_*) + \tau\sqrt{\frac{m-K_y\varepsilon_*}{M+\gamma_*}}\,(m-\gamma_*)\right]\|s_0\|^2 = K_{v^T s}\|s_0\|^2 > 0. \end{aligned}$$

Therefore the algorithm is well defined.

(d) To obtain the desired bounded deterioration property, we need to have an estimate of the form

$$\frac{\|M_\tau v_0 - M_\tau^{-1} s_0\|}{\|M_\tau^{-1} s_0\|} \leq K_v \sigma(x_0, x_1) \tag{3.19}$$

for the scale v_0 in our update (2.6) with a positive constant K_v (see (4.4) in [4]). Since by assumption a scale $v_0 = y_0 + \tau\rho_0 B_0^s s_0$ is chosen, the term

$$\rho_0 = \sqrt{\frac{y_0^T s_0}{s_0^T B_0^s s_0}} = \sqrt{1 + \frac{\|R(x_1)\|(y_0^{\#T} s_0 - s_0^T A_0 s_0)}{s_0^T B_0^s s_0}}$$

plays an important role in verifying (3.19) for our update strategy. We have to discuss two cases.

If $(y_0^{\#T} s_0 - s_0^T A_0 s_0) \geq 0$, then $\rho_0 \geq 1$, and we obtain

$$0 \leq \rho_0 - 1 \leq \frac{\|R(x_1)\|(y_0^{\#T} s_0 - s_0^T A_0 s_0)}{s_0^T B_0^s s_0}.$$

We can conclude with (3.3), (3.9), (3.14), and (3.17),

$$\frac{y_0^{\#T} s_0 - s_0^T A_0 s_0}{s_0^T B_0^s s_0} \leq \frac{|y_0^{\#T} s_0| + |s_0^T A_0 s_0|}{s_0^T B_0^s s_0} \leq \frac{L_J \frac{L_R}{\sqrt{\alpha}} + M(1+\tau)}{m-\gamma_*} = K^*. \tag{3.20}$$

This estimate therefore yields

$$0 \leq \rho_0 - 1 \leq L_R K^* \sigma(x_0, x_1). \tag{3.21}$$

For the second case, $y_0^{\#T} s_0 - s_0^T A_0 s_0 < 0$, we obtain, with assumption (A2), (3.5), and (3.13) together with the same estimates used in (3.20),

$$\begin{aligned} 1 > \rho_0 &= \sqrt{1 - \|R(x_1)\|\frac{s_0^T A_0 s_0 - y_0^{\#T} s_0}{s_0^T B_0^s s_0}} > 1 - \|R(x_1)\|\frac{s_0^T A s_0 - y_0^{\#T} s_0}{s_0^T B^s s_0} \\ &> 1 - L_R K^* \sigma(x_0, x_1) > 0. \end{aligned}$$

This implies

$$0 < 1 - \rho_0 < L_R K^* \sigma(x_0, x_1) \tag{3.22}$$

and therefore in either of the two cases we get

$$|\rho_0 - 1| \leq L_R K^* \sigma(x_0, x_1). \tag{3.23}$$

Since $y_0^T s_0 \leq (M + K_y \varepsilon_*)\|s_0\|^2$ by (3.15) and assumption (A3), with (3.17) we get

$$0 < \rho_0 < \frac{\sqrt{M + K_y \varepsilon_*}}{\sqrt{m - \gamma_*}}. \tag{3.24}$$

By the definition of M_τ in (3.7) the equation $M_\tau(1+\tau)B_* s_0 = M_\tau^{-1} s_0$ holds and we can use (3.15), (3.16), (3.23), and (3.24) to conclude

$$\begin{aligned}
\|M_\tau v_0 - M_\tau^{-1} s_0\| &= \|M_\tau v_0 - M_\tau(1+\tau)B_* s_0\| \\
&\leq \|M_\tau(y_0 - B_* s_0)\| + \tau \|M_\tau \rho_0 (B_0^s - B_*) s_0\| \\
&\quad + \tau |\rho_0 - 1| \|M_\tau B_* s_0\| \\
&\leq \frac{K_y}{\sqrt{m}\sqrt{1+\tau}} \sigma(x_0, x_1)\|s_0\| \\
&\quad + \frac{\tau\sqrt{M + K_y\varepsilon_*}}{\sqrt{m - \gamma_*}} \frac{(L_C + L_R(1+\tau)M)}{\sqrt{m}\sqrt{1+\tau}} \sigma(x_0, x_1)\|s_0\| \\
&\quad + \tau L_R K^* \frac{\sqrt{M}}{\sqrt{1+\tau}} \sigma(x_0, x_1)\|s_0\|.
\end{aligned}$$

Therefore we obtain

$$\frac{\|M_\tau v_0 - M_\tau^{-1} s_0\|}{\|M_\tau^{-1} s_0\|} \leq K_v \sigma(x_0, x_1) \tag{3.25}$$

with

$$K_v = \frac{K_y}{m(1+\tau)} + \tau \sqrt{\frac{M + K_y\varepsilon_*}{m - \gamma_*}} \frac{L_C + L_R(1+\tau)M}{m(1+\tau)} + \frac{\tau}{1+\tau}\sqrt{\frac{M}{m}} L_R K^*.$$

(e) By the results in the fundamental paper of Broyden, Dennis, and Moré [4], (see Lemma 5.2 there) it is now clear that for $A_0 \neq 0$, (3.19) implies the bounded deterioration property

$$\|A_1\|_{F,M_\tau} \leq (1 + \alpha_1 \sigma(x_0, x_1))\|A_0\|_{F,M_\tau} + \alpha_2 \sigma(x_0, x_1), \tag{3.26}$$

where

$$\alpha_1 = \frac{5}{2} \frac{K_v}{1 - K_v \varepsilon_*} \quad \text{and} \quad \alpha_2 = \frac{2(1 + 2\sqrt{n})}{\sqrt{m}} \|B_*^{-\frac{1}{2}}\|_F L_J \frac{L_R}{\sqrt{\alpha}} \Theta_* K,$$

while in the case $A_0 = 0$ direct calculations using (3.18), (3.25), and $\|v_0\| \leq K_{\|v\|}\|s_0\|$ show

$$\|A_1\|_{F,M_\tau} \leq \alpha_3 \sigma(x_0, x_1) \quad \text{with } \alpha_3 = \left(2 + \frac{K_{\|v\|}}{K_{v^T s}}\right) \frac{L_J L_R \Theta_* K_{\|v\|} K}{(1+\tau) m K_{v^T s}\sqrt{\alpha}}.$$

(3) In order to show

$$\|A_1\|_{F,M_\tau} < 1,$$

we use the boundedness arguments of Lemma 3.3 in [8] for matrices satisfying a bounded deterioration property.

If we define $\Gamma_i := \|A_i\|_{F,M_\tau}$ and $\delta_i := \max\{\alpha_1, \alpha_2\,\alpha_3\}\bar{\Theta}^i\varepsilon$, the bounded deterioration (3.26), together with the estimates in (3.12) for a previous iteration j for $0 \le j \le i-1$ in the case $i > 0$, implies

$$\Gamma_{j+1} \le (1+\delta_j)\Gamma_j + \delta_j \quad \text{for } 0 \le j \le i. \tag{3.27}$$

If we define $\mu_j := \prod_{k=0}^{j-1}(1+\delta_k)$, then we know for $0 \le j \le i+1 \quad (\mu_0 := 1)$,

$$\mu_j \ge 1 \quad \text{and} \quad \log \mu_j \le \sum_{k=0}^{j} \delta_k \le \max\{\alpha_1, \alpha_2\,\alpha_3\}\frac{\varepsilon}{1-\bar{\Theta}}.$$

These estimates imply $\mu_j \le e^{\max\{\alpha_1,\alpha_2,\alpha_3\}(\varepsilon/(1-\bar{\Theta}))}$.

Since with (3.27) we get

$$\frac{\Gamma_{j+1}}{\mu_{j+1}} \le \frac{\Gamma_j}{\mu_j} + \frac{\delta_j}{\mu_{j+1}} \le \frac{\Gamma_j}{\mu_j} + \delta_j, \tag{3.28}$$

we can use this argument iteratively to conclude that

$$\frac{\Gamma_{j+1}}{\mu_{j+1}} \le \Gamma_0 + \sum_{k=0}^{j} \delta_k \quad \text{for } 0 \le j \le i. \tag{3.29}$$

Hence

$$\Gamma_{i+1} \le e^{\max\{\alpha_1,\alpha_2,\alpha_3\}(\varepsilon/(1-\bar{\Theta}))}\left(\Gamma_0 + \max\{\alpha_1,\alpha_2,\alpha_3\}\frac{\varepsilon}{1-\bar{\Theta}}\right) < 1 \tag{3.30}$$

where the last estimate follows from (3.6).

(4) Now, since

$$\begin{aligned}\|B_1 - B_*\| &\le \|J(x_1)^T J(x_1) - J(x_*)^T J(x_*)\| + \|R(x_1)\|\|A_1\| \\ &\le L_C\|x_1 - x_*\| + L_R\|x_1 - x_*\|M(1+\tau) < \gamma,\end{aligned}$$

we can assure that the induction step can be made using identical arguments. □

Since the proof of Theorem 3.1 offers some interesting features, we will add some remarks.

Remark 3.1. If we choose $\tau = 1$ or $\tau = 0$ in Theorem 3.1, we obtain local q-quadratic convergence of the proposed self-scaling structured secant method (2.3)–(2.9) in the zero-residual case for the BFGS, respectively, the DFP, update.

Remark 3.2. Since $\|\cdot\|_{F,M_{\tau_1}} = \sqrt{(1+\tau_2)/(1+\tau_1)}\|\cdot\|_{F,M_{\tau_2}}$ the given proof can be extended to cover any update or variations of updates from the convex Broyden class. If we choose $\tau = 0$ in (3.1) and $\tau = 1$ in (3.2) and (3.6) and also in the estimates for K, K^*, K_v, the resulting constants $\tilde{\alpha}_1, \tilde{\alpha}_2$ in the bounded deterioration are independent of τ. If we also observe that (3.26) is equivalent to

$$\|A_1\|_* \le (1 + \tilde{\alpha}_1\sigma(x_0,x_1))\|A_0\|_* + \tilde{\alpha}_2\sqrt{1+\tau}\,\sigma(x_0,x_1), \tag{3.31}$$

we can cover all possible updates for $\tau \in [0,1]$ by substituting the maximum of $\tilde{\alpha}_1$ and $\sqrt{2}\tilde{\alpha}_2$ in (3.6). We have not given this general version, since for every τ the consideration of the special M_τ-norm is essential to derive the bounded deterioration property. We also want to mention that the constant 1 in the construction $\|A_0\|_{F,M_\tau} < 1$ can be changed to any other positive constant by changing the corresponding quantities in (3.1) and (3.6).

4. Q-superlinear convergence for nonzero-residual problems. Next we want to cover the nonzero-residual case. We start out with a lemma concerning the approximation property of $y^\#$ in the case of a nonzero-residual problem, since this property is fundamental for the convergence theory of structured secant methods (see [7], [13], and [18]).

LEMMA 4.1. *Suppose that the standard assumptions* (A1)–(A3) *for problem* (1.1) *hold with* $\|R(x_*)\| > 0$. *Then there exists a constant* $\tilde{K} > 0$ *such that*

$$\|y^\# - T(x_*)s\| \le \tilde{K}\sigma(x, x_+)\|s\| \quad \forall\, x, x_+ \in D_1 \tag{4.1}$$

with $s = x_+ - x$, $\sigma(x, x_+) = \max\{\|x - x_*\|, \|x_+ - x_*\|\}$, $y^\#$ *given in* (2.7), *and* $T(x)$ *defined in* (1.7).

Proof. First observe that $T(x_*)$ is well defined, since $\|R(x_*)\| > 0$.

$$\begin{aligned}
&\left\| (J(x_+) - J(x))^T \frac{R(x_+)}{\|R(x)\|} - \sum_{i=1}^m \frac{r_i(x_*)}{\|R(x_*)\|} \nabla^2 r_i(x_*)s \right\| \\
&\le \frac{1}{\|R(x)\|} \left\| (J(x_+) - J(x))^T R(x_+) - \sum_{i=1}^m r_i(x_*) \nabla^2 r_i(x_*)s \right\| \\
&\quad + \left(\frac{1}{\|R(x_*)\|} - \frac{1}{\|R(x)\|} \right) \left\| \sum_{i=1}^m r_i(x_*) \nabla^2 r_i(x_*)s \right\|.
\end{aligned}$$

Setting $\| \sum_{i=1}^m r_i(x_*) \nabla^2 r_i(x_*)\| =: K_2$ and since $\|R(x)\| \ge \|R(x_*)\|$ for all $x \in D_1$, assumption (A2) and the estimate

$$\left\| (J(x_+) - J(x))^T R(x_+) - \sum_{i=1}^m r_i(x_*) \nabla^2 r_i(x_*)s \right\| \le K_1 \sigma(x, x_+)\|s\| \tag{4.2}$$

given in Lemma 4.1 in [7], with a certain constant K_1, allow us to conclude that

$$\begin{aligned}
&\|y^\# - T(x_*)s\| \\
&\le \frac{K_1\sigma(x, x_+)\|s\|}{\|R(x)\|} + \frac{\|R(x)\| - \|R(x_*)\|}{\|R(x_*)\|\|R(x)\|} \cdot K_2\|s\| \\
&\le \frac{K_1}{\|R(x_*)\|}\sigma(x, x_+)\|s\| + \frac{\|R(x) - R(x_*)\|}{\|R(x_*)\|^2} K_2\|s\| \\
&\le \frac{K_1}{\|R(x_*)\|}\sigma(x, x_+)\|s\| + \frac{L_R\|x - x_*\|}{\|R(x_*)\|^2} K_2\|s\| \\
&\le \left(\frac{K_1}{\|R(x_*)\|} + \frac{K_2 L_R}{\|R(x_*)\|^2} \right) \sigma(x, x_+)\|s\|. \qquad \square
\end{aligned}$$

Following the convergence theory of structured secant methods developed in [7] and [19], we will establish the following lemma.

LEMMA 4.2. *Assume that the standard assumptions* (A1)–(A3) *hold with* $\|R(x_*)\| > 0$. *Then there exists a constant* $\tilde{K}^* > 0$ *such that*

$$\|y - \nabla^2 f(x_*)s\| \leq \tilde{K}^* \sigma(x, x_+)\|s\| \tag{4.3}$$

for y *given in* (2.8), $s = x_+ - x$ *and* $x, x_+ \in D_1$.

Proof. Since by the assumptions (A1) and (A2), with $K_3 = L_R\varepsilon_1 + \|R(x_*)\|$ for $y^\#$ in (2.7), we obtain

$$\|y^\#\| \leq L_J\|s\|\frac{\|R(x_+)\|}{\|R(x)\|} \leq L_J\|s\|\frac{K_3}{\|R(x_*)\|}.$$

Lemma 4.1 and the triangle inequality imply

$$\begin{aligned}
\|y - \nabla^2 f(x_*)s\| &\leq \|(J(x_+)^T J(x_+) - J(x_*)^T J(x_*))s\| \\
&\quad + \| \|R(x_+)\|y^\# - \|R(x_*)\|T(x_*)s\| \\
&\leq L_C\|x_+ - x_*\|\|s\| + \|R(x_*)\|\|y^\# - T(x_*)s\| \\
&\quad + (\|R(x_+)\| - \|R(x_*)\|)\|y^\#\| \\
&\leq L_C\|x_+ - x_*\| \|s\| + \|R(x_*)\|\tilde{K}\sigma(x, x_+)\|s\| + L_R\|x_+ - x_*\|\|y^\#\| \\
&\leq \tilde{K}^*\sigma(x, x_+)\|s\|
\end{aligned}$$

with $\tilde{K}^* = L_C + \tilde{K}\|R(x_*)\| + L_R L_J(K_3/\|R(x_*)\|)$. □

With these two lemmas we can establish the bounded deterioration property for the proposed method (2.3)–(2.9) in the nonzero-residual case.

THEOREM 4.3. *Suppose the standard assumptions* (A1)–(A3) *hold with* $\|R(x_*)\| > 0$ *and let* B_+ *be given by the structured secant update* (2.5); *i.e.,*

$$B_+ = J(x_+)^T J(x_+) + \|R(x_+)\|(A + \triangle_2(s, y^\#, A, v(s, y, B^s))),$$

where any scale v *is allowed such that*

$$v = y + \tau\sigma B^s s \quad \text{with } \tau \in [0, 1] \quad \text{and} \quad \sigma = \left(\frac{y^T s}{s^T B^s s}\right)^{\frac{1}{2}}.$$

Then there exist constants $\varepsilon > 0$ *and* $\delta > 0$, *such that* B_+ *is well defined and*

$$\|B_+ - \nabla^2 f(x_*)\|_* \leq [1 + \alpha_1\sigma(x, x_+)]\|B - \nabla^2 f(x_*)\|_* + \alpha_2\sigma(x, x_+) \tag{4.4}$$

holds for all $x, x_+ \in D_\varepsilon := \{x : \|x - x_*\| < \varepsilon\}$ *and for all* $A \in D_\delta := \{A : \|A - T(x_*)\| < \delta\}$.

Proof. Let ε, δ be so small that $B^s = J(x_+)^T J(x_+) + \|R(x_+)\|A$ is positive definite for $(x_+, A) \in D_\varepsilon \times D_\delta$, e.g., $(L_C + L_R\|T(x_*)\|)\varepsilon + (\|R(x_*)\| + L_R\varepsilon)\delta < m$.

Using observation (2.10), we conclude that

$$\begin{aligned}
\|B_+ - B_*\|_* &= \|B^s + \triangle_2(s, y, B^s, v) - B_*\|_* \\
&= \|B^s + \triangle_1(s, y, B^s, \phi) - B_*\|_*
\end{aligned} \tag{4.5}$$

for some $\phi \in [0, 1]$ by assumption on the scale v and Lemma 1.1.

Since B^s is positive definite and Lemma 4.2 assures the required assumption on y, the standard theory for the bounded deterioration property of unstructured updates

from the convex Broyden class (see Theorem 2.4 in [19]) implies with (4.6) for some nonnegative constants α_1, α_2:

$$\begin{aligned}\|B_+ - B_*\|_* &\le [1 + \alpha_1\sigma(x, x_+)]\,\|B^s - B_*\|_* + \alpha_2\sigma(x, x_+) \\ &\le [1 + \alpha_1\sigma(x, x_+)]\,(\|B - B_*\|_* + \|B^s - B\|_*) + \alpha_2\sigma(x, x_+).\end{aligned} \tag{4.6}$$

From the definition of B^s, the definition of the $\|\cdot\|_*$-norm, and the assumptions (A1) and (A2) we get

$$\begin{aligned}\|B^s - B\|_* &\le \|J(x_+)^T J(x_+) - J(x)^T J(x)\|_* + |\|R(x_+)\| - \|R(x)\|\|\,\|A\|_* \\ &\le \frac{\sqrt{n}}{m} L_C \|s\| + \frac{|\|R(x_+)\| - \|R(x)\|\|}{\|R(x)\|}\,\|R(x)\|\|A\|_* \\ &\le \frac{\sqrt{n}}{m} L_C \|s\| + \frac{L_R\|s\|}{\|R(x_*)\|}\,\|R(x)\|\|A\|_*.\end{aligned} \tag{4.7}$$

Adding the appropriate terms and using the triangle inequality we conclude that

$$\|R(x)\|\|A\|_* \le \|B - B_*\|_* + \frac{\sqrt{n}}{m} L_C \sigma(x, x_+) + \|R(x_*)\|\|T(x_*)\|_*.$$

Substituting this in (4.8), together with (4.7), yields

$$\|B_+ - B_*\|_* \le [1 + \bar{\alpha}_1\sigma(x, x_+)]\|B - B_*\| + \bar{\alpha}_2\sigma(x, x_+),$$

where

$$\bar{\alpha}_1 = \alpha_1 + (1 + \alpha_1\varepsilon)\frac{2L_R}{\|R(x_*)\|} \quad \text{and}$$

$$\bar{\alpha}_2 = \alpha_2 + (1 + \alpha_1\varepsilon)\left[\frac{2\sqrt{n}}{m} L_C + 2L_R\|T(x_*)\|_* + \frac{2\sqrt{n}L_C L_R\varepsilon}{m\|R(x_*)\|}\right]. \qquad \square$$

Using Theorem 4.3 in connection with the Broyden–Dennis–Moré theory of bounded deterioration [4] for secant methods, we can conclude the local q-linear convergence of the iterates x_k. But, moreover, we get local q-superlinear convergence of the totally structured secant method (2.3)–(2.9) for the nonzero-residual nonlinear least squares problem. In order to establish this result we will first show a theorem giving a sufficient condition for the Dennis–Moré condition of q-superlinear convergence. This lemma, which is implicitly contained in Proposition 4 in [16] and whose content was independently found for the special case of additive structure by [13], [17], and [18] offers a general tool for the analysis of quasi-Newton methods.

LEMMA 4.4. *Let $\{\varepsilon_k\}_{k=0}^{\infty}$ be a sequence in $\mathbb{R}$ with $\sum_{i=0}^{\infty}\varepsilon_k < \infty$, and let $\{x_k\}_{k=0}^{\infty}$ be a sequence in $\mathbb{R}^n$ such that $s_k := x_{k+1} - x_k \neq 0$ for all $k \in \mathbb{N}_0$. Let $B_* \in \mathbb{R}^{n\times n}$ be positive definite, $\{B_k\}_{k=0}^{\infty}$ a sequence of positive-definite real $n \times n$-matrices such that the sequence of condition numbers $\{\mathrm{cond}_2(B_k)\}_{k=0}^{\infty}$ is bounded. Also let $\{\phi_k\}_{k=1}^{\infty}$ be a sequence in $[0, 1]$, such that for*

$$B_k' = B_k + \triangle_1(s_k, B_* s_k, B_k, \phi_k) \tag{4.8}$$

in the weighted Frobenius norm $\|\cdot\|_*$ induced by B_*, the inequality

$$\|B_{k+1} - B_k'\|_* \le \bar{\alpha}_1\varepsilon_k \tag{4.9}$$

holds for all $k \in \mathbb{N}$ with some nonnegative constant $\bar{\alpha}_1$. Then it follows that

$$\lim_{k\to\infty} \frac{\|(B_k - B_*)s_k\|}{\|s_k\|} = 0.$$

Proof. If we define $\bar{B}'_k := B_*^{-\frac{1}{2}} B'_k B_*^{-\frac{1}{2}}$ and also $\bar{B}_k := B_*^{-\frac{1}{2}} B_k B_*^{-\frac{1}{2}}$ we then have

$$\|B'_k - B_*\|_*^2 = \|\bar{B}'_k - I\|_F^2; \qquad \|B_k - B_*\|_*^2 = \|\bar{B}_k - I\|_F^2. \tag{4.10}$$

With $\tilde{v}_k := B_*^{\frac{1}{2}} s_k$, by Lemma 3.3 in [19], it follows that

$$\|B'_k - B_*\|_*^2 - \|B_k - B_*\|_*^2 = -p_k - 2q_k\phi_k + r_k\phi_k^2 \le 0, \tag{4.11}$$

where p_k, q_k, r_k are defined by

$$\begin{aligned} p_k &= (1 - z_k^T z_k)^2 + 2\{z_k^T \bar{B}_k z_k - (z_k^T z_k)^2\} \\ q_k &= (z_k^T z_k)^2 + z_k^T z_k - (\alpha_k^2 + z_k^T \bar{B}_k z_k) \\ r_k &= (z_k^T w_k)^2, \end{aligned} \tag{4.12}$$

with $u_k = \tilde{v}_k/\|\tilde{v}_k\|, z_k = \bar{B}_k\tilde{v}_k/\sqrt{\tilde{v}_k^T \bar{B}'_k \tilde{v}_k}, \alpha_k = u_k^T z_k$, and $w_k = \alpha_k u_k - z_k$. With these definitions one can verify the identity

$$\begin{aligned} -p_k - 2q_k\phi_k + r_k\phi_k^2 = &-[(1-\phi_k)p_k] \\ &-[\phi_k\{(1-\alpha_k^2)^2 + 2\phi_k\alpha_k^2(z_k^T z_k - \alpha_k^2)\}] \\ &-[\phi_k(1-\phi_k)\{z_k^T z_k^2 - \alpha_k^4\}]. \end{aligned} \tag{4.13}$$

Since by the Cauchy–Schwarz inequality $z_k^T z_k \ge \alpha_k^2$ and Lemma 3.3 in [19] ensures $p_k \ge 0$, all three parts in the squared brackets are nonnegative. The assumptions on the sequence $\{B_k\}_{k=0}^{\infty}$, together with the inequality (4.11), imply the existence of some constant c_1 such that for all $k \in \mathbb{N}_0$ the following holds:

$$\|B'_k - B_*\|_* \le \|B_k - B_*\|_* \le c_1. \tag{4.14}$$

Using (4.9) and (4.11), we conclude that

$$\begin{aligned} 0 &\le \sum_{k=0}^{\infty} (\|B_k - B_*\|_*^2 - \|B'_k - B_*\|_*^2) \\ &\le \sum_{k=0}^{\infty} (\|B_k - B_*\|_*^2 - \|B_{k+1} - B_*\|_*^2 + 2\bar{\alpha}_1 c_1 \varepsilon_k) \\ &\le \|B_0 - B_*\|_*^2 + 2\bar{\alpha}_1 c_1 \sum_{k=0}^{\infty} \varepsilon_k < \infty. \end{aligned}$$

Therefore

$$\lim_{k\to\infty} (\|B_k - B_*\|_*^2 - \|B'_k - B_*\|_*^2) = \lim_{k\to\infty} (-p_k - 2q_k\phi_k + r_k\phi_k^2) = 0. \tag{4.15}$$

By the remarks following (4.14), therefore, all three parts of the sum in (4.14) converge to 0. In the further analysis, which is close to the proof of Proposition 4 in [16], we must consider three different cases.

(i) $\limsup_{k\to\infty}\phi_k < 1$. This implies $1-\phi_k > \chi > 0$ for a certain χ and $k \geq k_0$. Since $(1-\phi_k)p_k \to 0$, we therefore have $p_k \to 0$. Using the definition of p_k in (4.13) and the inequalities (see the proof of Lemma 3.3 in [19])

$$z_k^T z_k \geq \alpha_k^2 \quad \text{and} \quad z_k^T \bar{B}_k z_k \geq z_k^T z_k^2 \tag{4.16}$$

it follows that

$$z_k^T z_k = \frac{\tilde{v}_k^T \bar{B}_k^2 \tilde{v}_k}{\tilde{v}_k^T \bar{B}_k \tilde{v}_k} \to 1 \quad \text{and} \quad z_k^T \bar{B}_k z_k = \frac{\tilde{v}_k^T \bar{B}_k^3 \tilde{v}_k}{\tilde{v}_k^T \bar{B}_k \tilde{v}_k} \to 1. \tag{4.17}$$

Hence we get

$$\lim_{k\to\infty} \frac{\|\bar{B}_k^{\frac{1}{2}}(\bar{B}_k - I)\tilde{v}_k\|_2^2}{\|\bar{B}_k^{\frac{1}{2}}\tilde{v}_k\|_2^2} = \lim_{k\to\infty} \frac{\tilde{v}_k^T \bar{B}_k \tilde{v}_k + \tilde{v}_k^T \bar{B}_k^3 \tilde{v}_k - 2\tilde{v}_k^T \bar{B}_k^2 \tilde{v}_k}{\tilde{v}_k^T \bar{B}_k \tilde{v}_k} = 0. \tag{4.18}$$

Since $\{\mathrm{cond}_2(B_k)\}_{k=0}^{\infty}$ is bounded and $\|\bar{B}_k^{-\frac{1}{2}} y\|_2 / \|\bar{B}_k^{-\frac{1}{2}} x\|_2 \leq \mathrm{cond}_2(\bar{B}_k^{\frac{1}{2}})(\|y\|_2/\|x\|_2)$ for all $x, y \in \mathbb{R}^n - \{0\}$, (4.18), together with the equivalence of norms on finite-dimensional spaces and the definition of $\tilde{v}_k$, implies

$$0 = \lim_{k\to\infty} \frac{\|(\bar{B}_k - I)\tilde{v}_k\|_2}{\|\tilde{v}_k\|_2} = \lim_{k\to\infty} \frac{\|(B_k - B_*)s_k\|}{\|s_k\|}. \tag{4.19}$$

(ii) $\liminf_{k\to\infty}\phi_k > 0$ By (4.15), (4.14), and the remarks following (4.14), this implies

$$\phi_k((1-\alpha_k^2)^2 + 2\phi_k\alpha_k^2(z_k^T z_k - \alpha_k^2)) \to 0 \quad (k \to \infty).$$

Since $z_k^T z_k \geq \alpha_k^2$ we can conclude that

$$\alpha_k^2 = \frac{\tilde{v}_k^T \bar{B}_k \tilde{v}_k}{\tilde{v}_k^T \tilde{v}_k} \to 1. \quad \text{Hence } z_k^T z_k = \frac{\tilde{v}_k^T \bar{B}_k^2 \tilde{v}_k}{\tilde{v}_k^T \bar{B}_k \tilde{v}_k} \to 1 \quad \text{and} \quad \frac{\tilde{v}_k^T \bar{B}_k^2 \tilde{v}_k}{\tilde{v}_k^T \tilde{v}_k} \to 1. \tag{4.20}$$

This, together with the equivalence of norms in finite-dimensional spaces, implies

$$\begin{aligned} \lim_{k\to\infty} \frac{\|(B_k - B_*)s_k\|_2^2}{\|s_k\|_2^2} &= \lim_{k\to\infty} \frac{\|(\bar{B}_k - I)\tilde{v}_k\|_2^2}{\|\tilde{v}_k\|_2^2} \\ &= \lim_{k\to\infty} \frac{\tilde{v}_k^T \tilde{v}_k - 2\tilde{v}_k^T \bar{B}_k \tilde{v}_k + \tilde{v}_k^T \bar{B}_k^2 \tilde{v}_k}{\tilde{v}_k^T \tilde{v}_k} = 0. \end{aligned}$$

(iii) If neither $\limsup_{k\to\infty}\phi_k < 1$ nor $\liminf_{k\to\infty}\phi_k > 0$, we can apply the above arguments to the partial sequences $\{\phi_{k_n} : k_n \in \mathbb{N}; \phi_{k_n} \leq \frac{1}{2}\}$ and $\{\phi_{k_n} : k_n \in \mathbb{N}; \phi_{k_n} > \frac{1}{2}\}$. □

THEOREM 4.5. *Suppose that the standard assumptions* (A1)–(A3) *hold in the nonzero-residual case* $\|R(x_*)\| > 0$. *Then there exist* $\varepsilon, \delta > 0$, *such that for* $\|x_0 - x_*\| < \varepsilon$ *and* $A_0 \in \mathbb{R}^{n\times n}$ *symmetric with* $\|A_0 - T(x_*)\| < \delta$ *the sequence of iterates* x_k, *generated by the* TSSM (2.3)–(2.9) *using a scale of the form*

$$v_k = y_k + \tau_k \rho_k B_k^s s_k \quad \textit{with } \tau_k \in [0,1] \quad \textit{and} \quad \rho_k = \left(\frac{y_k^T s_k}{s_k^T B_k^s s_k}\right)^{\frac{1}{2}},$$

is well defined and converges q-superlinearly to x_*.

Proof. Theorem 4.3, together with the Broyden–Dennis–Moré theory [4], ensures the q-linear convergence of the sequence $\{x_k\}_{k=0}^{\infty}$, the boundedness of the sequences $\{B_k\}_{k=0}^{\infty}$ and $\{B_k^{-1}\}_{k=0}^{\infty}$, and the positive-definiteness of B_k and B_k^s for sufficiently small ε and δ. In order to establish the q-superlinear convergence, we will use the content of Lemma 4.4 with $B_* := \nabla^2 f(x_*)$. With the definition $B_k' := B_k + \triangle_1(s_k, B_* s_k, B_k, \phi_k)$ (where ϕ_k corresponds to the scale v_k) we will verify the estimate

$$\|B_{k+1} - B_k'\|_* \leq (\alpha_1 \|B_k - B_*\|_* + \alpha_2)\sigma(x_k, x_{k+1}) \leq \alpha\sigma(x_k, x_{k+1}) \tag{4.21}$$

for certain constants $\alpha_1, \alpha_2, \alpha$ in order to establish the Dennis–Moré characterization [8] for the proposed method, since the Broyden–Dennis–Moré theory ensures that all other assumptions of Lemma 4.4 are fulfilled if it is applied to the sequences $\{x_k\}_{k=0}^{\infty}$ and $\{B_k\}_{k=0}^{\infty}$.

In order to verify (4.21) for our self-scaling structured secant method, with $\bar{B}_{k+1} = B_k + \triangle_1(s_k, y_k, B_k, \phi_k)$, we conclude that

$$\|B_{k+1} - B_k'\|_* \leq \|B_{k+1} - \bar{B}_{k+1}\|_* + \|\bar{B}_{k+1} - B_k'\|_*.$$

(a) Since $\bar{B}_{k+1}$ is an unstructured update from the convex class and Lemma 4.2 ensures $\|y - B_* s\| \leq \tilde{K}\sigma(x, x_+)\|s\|$, a result of Martínez (see the proof of Theorem 2.4 in [19]) ensures

$$\|\bar{B}_{k+1} - B_k'\|_* \leq (\tilde{\alpha}_1 \|B_k - B_*\|_* + \tilde{\alpha}_2)\sigma(x_k, x_{k+1})$$

for appropriate constants $\tilde{\alpha}_1$ and $\tilde{\alpha}_2$.

(b) For the remaining expression with (2.10), we get

$$\begin{aligned}\|B_{k+1} - \bar{B}_{k+1}\|_* &= \|B_k^s + \triangle_1(s_k, y_k, B_k^s, \phi_k) - B_k - \triangle_1(s_k, y_k, B_k, \phi_k)\|_* \\ &\leq \|B_k^s - B_k\|_* + \frac{1}{m}\|\triangle_1(s_k, y_k, B_k^s, \phi_k) - \triangle_1(s_k, y_k, B_k, \phi_k)\|_F.\end{aligned}$$

(i) Using the definition of B^s, the assumptions (A1) and (A2), together with some properties of the weighted Frobenius norm $\|\cdot\|_*$, yield

$$\begin{aligned}\|B_k^s - B_k\|_* &= \|J(x_{k+1})^T J(x_{k+1}) - J(x_k)^T J(x_k) + (\|R(x_{k+1})\| - \|R(x_k)\|)A_k\|_* \\ &\leq \frac{\sqrt{n}L_C}{m}\|s_k\| + \frac{|\,\|R(x_{k+1})\| - \|R(x_k)\|\,|}{\|R(x_k)\|}\|R(x_k)\|\|A_k\|_* \\ &\leq \frac{\sqrt{n}L_C}{m}\|s_k\| + \frac{2L_R}{\|R(x_*)\|}\sigma(x_k, x_{k+1})\left[\|B_k - B_*\|_*\right. \\ &\qquad \left. + \frac{\sqrt{n}L_C}{m}\varepsilon + \|R(x_*)\|\|T(x_*)\|_*\right] \\ &\leq (\bar{\alpha}_1 \|B_k - B_*\|_* + \bar{\alpha}_2)\sigma(x_k, x_{k+1})\end{aligned}$$

with the appropriate constants $\bar{\alpha}_1$, and $\bar{\alpha}_2$.

(ii) An easy calculation using the definition of the $\triangle_1$-update in (1.10) and some properties of the trace of a matrix shows

$$\begin{aligned}&\|\triangle_1(s_k, y_k, B_k^s, \phi_k) - \triangle_1(s_k, y_k, B_k, \phi_k)\|_F \\ &\leq [(1 - \phi_k)(2L_1 + L_1^2) + \phi_k(2L_2 + L_2^2)]\|B_k^s - B_k\|_2,\end{aligned}$$

for $L_1 \geq \max\{\text{cond}_2(B_k), \text{cond}_2(B_k^s)\}$, and $L_2 \geq \|y_k\|\|s_k\|/y_k^T s_k$ (compare with Lemma 3.8 in [18]).

Therefore we need to find estimates for L_1 and L_2, which are independent of k. Since $\{B_k\}$ and $\{B_k^{-1}\}$ are bounded according to the Broyden–Dennis–Moré theory, say $\|B_k\| \leq c_1$ and $\|B_k^{-1}\| \leq c_2$, we know from the inequality in (i) and the q-linear convergence of the x_k, that $\text{cond}_2(B_k^s)$ is also bounded, $\|B_k^s\| \leq \bar{c}_1$; $\|(B_k{}^s)^{-1}\| \leq \bar{c}_2$. So we can choose $L_1 = \max\{c_1 \cdot c_2, \bar{c}_1 \cdot \bar{c}_2\}$.

The estimate L_2 for $\|y_k\|\|s_k\|/y_k^T s_k$ follows directly from Proposition 2.1 in [7], since Lemma 4.2 ensures the required condition with a constant $\tilde{K}^*$.

(iii) With $L^* = \max\{L_1, L_2\}$ we can summarize the results from (i) and (ii) in the inequality

$$\|B_{k+1} - \bar{B}_{k+1}\|_* \leq (\alpha_1^* \|B_k - B_*\|_* + \alpha_2^*)\sigma(x_k, x_{k+1}),$$

where

$$\alpha_1^* = \bar{\alpha}_1 \left(1 + \frac{(2L^* + L^{*^2})M}{m}\right) \quad \text{and} \quad \alpha_2^* = \bar{\alpha}_2 \left(1 + \frac{(2L^* + L^{*^2})M}{m}\right).$$

Combining (a) with (b) and using the boundedness of $\|B_k\|$, we have shown (4.21) with $\alpha_i = \tilde{\alpha}_i + \bar{\alpha}_i$, $i = 1, 2$.

(c) By Lemma 4.4 (4.21) therefore implies the Dennis–Moré condition

$$\lim_{k \to \infty} \frac{\|(B_k - \nabla^2 f(x_*))s_k\|}{\|s_k\|} = 0$$

and hence the local q-superlinear convergence. □

5. Numerical examples. In this section we want to discuss the numerical behavior of the proposed TSSM for nonlinear least squares problems (2.3)–(2.9). Since TSSM is intended as a local method, at this point we do not want to discuss matters of globalization, but concentrate on the local behavior in comparison to the Gauss–Newton method and the structured secant method proposed in [6] and [7]. Since most of the starting points in the standard test problems for nonlinear least squares solvers [20] focus on the globalization, in our examples, reflecting the aspects of the local behavior, we will start out from points where globalization is usually less important and full step length is acceptable. The following calculations were done in double precision Fortran on a SUN Sparcstation 1, and in all examples we use $A_0 = I$ and $\tau = 1$. The notation $e_i = x_i - x_*$ will be used to denote the error, and the ratios $\|e_i\|/\|e_{i-1}\|$ are given to show the rates of convergence.

In Examples 5.1 and 5.2 we show the behavior of TSSM on zero-residual problems. For comparison we will also give the iterates of the Gauss–Newton method. Comparing these results it is apparent that in the first iterations the use of the approximations to the second-order part gives better directions, while closer to the solution the self-adjusting effect induces the q-quadratic convergence. Therefore the main idea of the BFGS-Gauss–Newton-hybrid method [1], [15] is present in this algorithm, but the problem of the appropriate switching is avoided.

Example 5.1 (Helical valley function).

$$r_1(x) = 10[x_3 - 10\Psi(x_1, x_2)], \quad r_2(x) = 10[(x_1^2 + x_2^2)^{\frac{1}{2}} - 1], \quad r_3(x) = x_3,$$

where

$$\Psi(x_1, x_2) = \begin{cases} \frac{1}{2\pi} \arctan\left(\frac{x_2}{x_1}\right) : & x_1 > 0, \\ \frac{1}{2\pi} \arctan\left(\frac{x_2}{x_1}\right) + 0.5 : & x_1 < 0. \end{cases}$$

Starting point: $x_0 = (6, 4, -2), \|\nabla f(x_0)\| = 690.09.$

TABLE 5.1
Zero-residual problem; comparison to Gauss–Newton method.

HELICAL VALLEY FUNCTION				
	TSSM		Gauss–Newton	
iter	$\|\nabla f(x)\|$	$\|e_i\|/\|e_{i-1}\|$	$\|\nabla f(x)\|$	$\|e_i\|/\|e_{i-1}\|$
1	$0.2809996 * 10^{3}$	$0.40568 * 10^{0}$	$0.3589708 * 10^{3}$	$0.54991 * 10^{0}$
2	$0.8200570 * 10^{2}$	$0.24212 * 10^{0}$	$0.2590070 * 10^{3}$	$0.70581 * 10^{0}$
3	$0.2697109 * 10^{1}$	$0.13886 * 10^{0}$	$0.1509635 * 10^{3}$	$0.49816 * 10^{0}$
4	$0.1815065 * 10^{0}$	$0.33157 * 10^{0}$	$0.8460491 * 10^{2}$	$0.46836 * 10^{0}$
5	$0.1232658 * 10^{-1}$	$0.42394 * 10^{-1}$	$0.3479522 * 10^{2}$	$0.25596 * 10^{0}$
6	$0.2925564 * 10^{-4}$	$0.48059 * 10^{-2}$	$0.3196235 * 10^{1}$	$0.86254 * 10^{-1}$
7	$0.5276053 * 10^{-9}$	$0.11175 * 10^{-4}$	$0.2668614 * 10^{-1}$	$0.76788 * 10^{-2}$
8			$0.1477139 * 10^{-5}$	$0.59296 * 10^{-4}$
9			$0.5443261 * 10^{-14}$	$0.29791 * 10^{-8}$

Example 5.2 (Beale function).

$$r_1(x) = 1.5 - x_1(1 - x_2), r_2(x) = 2.25 - x_1(1 - {x_2}^2),$$
$$r_3(x) = 2.625 - x_1(1 - {x_2}^3).$$

Starting Point: $x_0 = (2.9, 0.9), \|\nabla f(x_0)\| = 25.35$

TABLE 5.2
Zero-residual problem; comparison to Gauss–Newton method.

BEALE TEST FUNCTION				
	TSSM		Gauss–Newton	
iter	$\|\nabla f(x)\|$	$\|e_i\|/\|e_{i-1}\|$	$\|\nabla f(x)\|$	$\|e_i\|/\|e_{i-1}\|$
1	$0.3957538 * 10^{1}$	$0.35175 * 10^{0}$	$0.2157703 * 10^{2}$	$0.14599 * 10^{1}$
2	$0.2601626 * 10^{0}$	$0.17680 * 10^{0}$	$0.1810082 * 10^{2}$	$0.14478 * 10^{1}$
3	$0.6623799 * 10^{-2}$	$0.32353 * 10^{0}$	$0.1461573 * 10^{2}$	$0.13342 * 10^{1}$
4	$0.1929555 * 10^{-3}$	$0.18363 * 10^{-1}$	$0.1087433 * 10^{2}$	$0.12647 * 10^{1}$
5	$0.9880714 * 10^{-7}$	$0.12436 * 10^{-3}$	$0.4979415 * 10^{1}$	$0.13340 * 10^{1}$
6	$0.484952 * 10^{-14}$	$0.24160 * 10^{-7}$	$0.7967627 * 10^{-1}$	$0.17264 * 10^{-2}$
7			$0.2258791 * 10^{-3}$	$0.30535 * 10^{-1}$
8			$0.1810402 * 10^{-7}$	$0.22429 * 10^{-4}$

In Examples 5.1 and 5.2 the nonlinearities had some impact on the iteration. In cases of zero-residual problems with mild nonlinearities the Gauss–Newton method is superior, although due to the quadratic convergence the TSSM does not lose much.

In a second part we give examples of TSSM in comparison to the original structured secant method proposed in [6] and [7]. In the numerical examples, we use the

structured secant method in the prescaled version described in [11], with the prescaling quantity $\min\{1, y^T s/s^T As\}$. In Example 5.3 we compare the behavior of these two methods on a zero-residual problem. Example 5.4 is a parametrized function, where the parameter Ψ can be used to control the size of the residual. While in the case of nonzero-residual problems the affinity of TSSM to the structured secant method results in similar numerical behavior, the controlled way of resizing the second-order approximations shows its payoff against the strategy of prescaled updates in zero-residual problems.

Example 5.3 (Freudenstein and Roth function).

$$r_1(x) = -13 + x_1 + ((5 - x_2)x_2 - 2)x_2, r_2(x) = -29 + x_1 + ((x_2 + 1)x_2 - 14)x_2.$$

Starting point: $x_0 = (4, 4), \|\nabla f(x_0)\| = 32.06.$

TABLE 5.3
Zero-residual problem; comparison to structured secant method.

FREUDENSTEIN–ROTH TEST FUNCTION				
	TSSM		Structured Secant Method	
iter	$\|\nabla f(x)\|$	$\|e_i\|/\|e_{i-1}\|$	$\|\nabla f(x)\|$	$\|e_i\|/\|e_{i-1}\|$
1	$0.7192578 * 10^{0}$	$0.49364 * 10^{0}$	$0.7192578 * 10^{0}$	$0.4936427 * 10^{0}$
2	$0.2215899 * 10^{1}$	$0.10387 * 10^{0}$	$0.3096185 * 10^{1}$	$0.14541 * 10^{0}$
3	$0.3669738 * 10^{-1}$	$0.16394 * 10^{-2}$	$0.4790255 * 10^{0}$	$0.19409 * 10^{-1}$
4	$0.1784708 * 10^{-5}$	$0.26325 * 10^{-3}$	$0.2006965 * 10^{-1}$	$0.19484 * 10^{0}$
5	$0.8817841 * 10^{-15}$	$0.40134 * 10^{-7}$	$0.1175471 * 10^{-3}$	$0.44249 * 10^{-1}$
6			$0.4662133 * 10^{-5}$	$0.11037 * 10^{-1}$
7			$0.4416952 * 10^{-11}$	$0.85809 * 10^{-6}$

Example 5.4 (Parametrized test function).

$$r_1(x) = x_1 - 2, \quad r_2(x) = (x_1 - 2\Psi)x_2, \quad r_3(x) = x_2 + 1.$$

Zero-residual problem for $\Psi = 1$; nonzero-residual problem for $\Psi \neq 1$.

Starting point: $x_0 = (1.1, 0.6)$.

$$\Psi = 1 : \|\nabla f(x_0)\| = 2.74; \qquad \Psi = 0.5 : \|\nabla f(x_0)\| = 2.14;$$
$$\Psi = 0 : \|\nabla f(x_0)\| = 2.55.$$

TABLE 5.4
Convergence of the approximation matrices.

PARAMETRIZED TEST FUNCTION $\Psi = 1$ (Ex. 3.6.4)		
	TSSM	Structured Secant Method
iter	$\|R(x_i)\|\|A_i\|_F/(\|R(x_{i-1})\|\|A_{i-1}\|_F)$	$\gamma_i\|A_i\|_F/(\gamma_{i-1}\|A_{i-1}\|_F)$
1	$0.470 * 10^{0}$	$0.467 * 10^{0}$
2	$0.412 * 10^{0}$	$0.113 * 10^{0}$
3	$0.168 * 10^{0}$	$0.780 * 10^{0}$
4	$0.270 * 10^{-1}$	$0.134 * 10^{0}$
5	$0.153 * 10^{-3}$	$0.868 * 10^{-1}$
6	$0.542 * 10^{-5}$	$0.889 * 10^{0}$
7		$0.189 * 10^{-1}$
8		$0.228 * 10^{-1}$

TABLE 5.5
Zero- and nonzero-residual problem; comparison to structured secant method.

PARAMETRIZED TEST FUNCTION				
	TSSM		Structured Secant Method	
$\Psi = 1$ Zero-residual problem				
iter	$\|\nabla f(x)\|$	$\|e_i\|/\|e_{i-1}\|$	$\|\nabla f(x)\|$	$\|e_i\|/\|e_{i-1}\|$
1	$0.1208570 * 10^{1}$	$0.67713 * 10^{0}$	$0.1208570 * 10^{1}$	$0.67713 * 10^{0}$
2	$0.3573220 * 10^{0}$	$0.39773 * 10^{0}$	$0.3911876 * 10^{0}$	$0.42293 * 10^{0}$
3	$0.8079856 * 10^{-1}$	$0.16851 * 10^{0}$	$0.1722076 * 10^{0}$	$0.30093 * 10^{0}$
4	$0.3551647 * 10^{-2}$	$0.42531 * 10^{-1}$	$0.6896362 * 10^{-1}$	$0.40725 * 10^{0}$
5	$0.1066625 * 10^{-4}$	$0.30100 * 10^{-2}$	$0.8550744 * 10^{-2}$	$0.13171 * 10^{0}$
6	$0.5788237 * 10^{-10}$	$0.54267 * 10^{-5}$	$0.7721650 * 10^{-4}$	$0.90983 * 10^{-2}$
7			$0.5166604 * 10^{-6}$	$0.66914 * 10^{-2}$
8			$0.1101534 * 10^{-9}$	$0.21320 * 10^{-3}$
$\Psi = 0.5$ Nonzero-residual problem				
iter	$\|\nabla f(x)\|$	$\|e_i\|/\|e_{i-1}\|$	$\|\nabla f(x)\|$	$\|e_i\|/\|e_{i-1}\|$
1	$0.1139912 * 10^{1}$	$0.57914 * 10^{0}$	$0.1139912 * 10^{1}$	$0.57914 * 10^{0}$
2	$0.4460727 * 10^{0}$	$0.22942 * 10^{0}$	$0.3785536 * 10^{0}$	$0.18336 * 10^{0}$
3	$0.4845131 * 10^{-1}$	$0.29635 * 10^{0}$	$0.9498244 * 10^{-1}$	$0.49028 * 10^{0}$
4	$0.1426845 * 10^{-1}$	$0.28730 * 10^{0}$	$0.2472530 * 10^{-1}$	$0.38463 * 10^{0}$
5	$0.6981968 * 10^{-4}$	$0.31283 * 10^{-2}$	$0.6679699 * 10^{-2}$	$0.16416 * 10^{0}$
6	$0.1788921 * 10^{-5}$	$0.12492 * 10^{-1}$	$0.6984975 * 10^{-3}$	$0.54627 * 10^{-1}$
7	$0.6495020 * 10^{-7}$	$0.37499 * 10^{-1}$	$0.2146816 * 10^{-4}$	$0.45128 * 10^{-1}$
8	$0.2415010 * 10^{-8}$	$0.17825 * 10^{-3}$	$0.3171587 * 10^{-6}$	$0.29333 * 10^{-1}$
9			$0.9056724 * 10^{-8}$	$0.27307 * 10^{-1}$
10			$0.3437382 * 10^{-9}$	$0.95183 * 10^{-2}$
$\Psi = 0.0$ Nonzero-residual problem				
iter	$\|\nabla f(x)\|$	$\|e_i\|/\|e_{i-1}\|$	$\|\nabla f(x)\|$	$\|e_i\|/\|e_{i-1}\|$
1	$0.1105368 * 10^{1}$	$0.35686 * 10^{0}$	$0.1105368 * 10^{1}$	$0.35686 * 10^{0}$
2	$0.2582951 * 10^{0}$	$0.20684 * 10^{0}$	$0.2129625 * 10^{0}$	$0.21267 * 10^{0}$
3	$0.2332925 * 10^{-1}$	$0.37795 * 10^{0}$	$0.2815997 * 10^{-1}$	$0.43741 * 10^{0}$
4	$0.1113643 * 10^{-1}$	$0.50877 * 10^{0}$	$0.1146481 * 10^{-1}$	$0.41134 * 10^{0}$
5	$0.9157531 * 10^{-3}$	$0.45014 * 10^{-1}$	$0.1601301 * 10^{-2}$	$0.63549 * 10^{-1}$
6	$0.4639255 * 10^{-4}$	$0.18689 * 10^{-1}$	$0.1127188 * 10^{-3}$	$0.33780 * 10^{-1}$
7	$0.1206947 * 10^{-5}$	$0.47388 * 10^{-1}$	$0.3424321 * 10^{-5}$	$0.69244 * 10^{-1}$
8	$0.4669132 * 10^{-7}$	$0.37445 * 10^{-1}$	$0.1801885 * 10^{-6}$	$0.52045 * 10^{-1}$
9	$0.1800063 * 10^{-8}$	$0.22337 * 10^{-3}$	$0.2968174 * 10^{-7}$	$0.18269 * 10^{0}$
10			$0.6308172 * 10^{-8}$	$0.14004 * 10^{0}$

Acknowledgments. The author owes thanks to Prof. E. Sachs for his valuable suggestions and his encouraging remarks.

REFERENCES

[1] M. AL-BAALI AND R. FLETCHER, *Variational methods for nonlinear least squares*, J. Oper. Res. Soc., 36 (1985), pp. 405–421.

[2] M. C. BARTHOLOMEW-BIGGS, *The estimation of the Hessian matrix in nonlinear least squares problems with non-zero residuals*, Math. Programming, 12 (1977), pp. 67–80.

[3] A. BJÖRCK, *Least squares methods*, in Handbook of Numerical Analysis, Vol.1: Solution of

Equations in $\mathbb{R}^n$, P. Ciarlet and J. Lions, eds., Elsevier/North-Holland, Amsterdam, 1989, pp. 469–647.

[4] C. G. BROYDEN, J. E. DENNIS, AND J. J. MORÉ, *On the local and superlinear convergence of quasi-Newton methods*, J. IMA, 12 (1973), pp. 223–245.

[5] J. E. DENNIS, *Nonlinear least squares and equations*, in The State of the Art in Numerical Analysis, D. Jacobs, ed., Academic Press, New York, 1977, pp. 269–312.

[6] J. E. DENNIS, D. M. GAY, AND R. E.WELSCH, *An adaptive nonlinear least-squares algorithm*, Trans. Math. Software, 7 (1981), pp. 348–368.

[7] J. E. DENNIS, H. J. MARTÍNEZ, AND R. A. TAPIA, *Convergence theory for the structured BFGS secant method with an application to nonlinear least squares*, J. Optim. Theory Appl., 61 (1989), pp. 161–178.

[8] J. E. DENNIS AND J. J. MORÉ, *A characterization of superlinear convergence and its application to quasi-Newton methods*, Math. Comp., 28 (1974), pp. 549–560.

[9] ———, *Quasi-Newton methods, motivation and theory*, SIAM Rev., 19 (1977), pp. 46–89.

[10] J. E. DENNIS AND R. B. SCHNABEL, *Least change secant updates for quasi-Newton methods*, SIAM Rev., 21 (1979), pp. 443–459.

[11] ———, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, Englewood Cliffs, NJ, 1983.

[12] J. E. DENNIS AND H. F. WALKER, *Convergence theorems for least change secant update methods*, SIAM J. Numer. Anal., 18 (1981), pp. 949–987.

[13] J. ENGELS AND H. MARTINEZ, *Local and superlinear convergence for partially known quasi-Newton methods*, SIAM J. Optim., 1 (1991), pp. 42–56.

[14] R. FLETCHER, *Practical Methods of Optimization*, John Wiley and Sons, New York, 1987.

[15] R. FLETCHER AND C. XU, *Hybrid methods for nonlinear least squares*, IMA J. Numer. Anal., 7 (1987), pp. 371–389.

[16] A. GRIEWANK AND P. L. TOINT, *Local convergence analysis for partitioned quasi-Newton methods*, Numer. Math., 39 (1982), pp. 429–448.

[17] J.HUSCHENS, *Exploiting additional structure in equality constrained optimizati on by structured SQP secant algorithms*, J. Optim. Theory Appl., 77 (1993), pp. 343–359.

[18] ———, *Konvergenztheorie und Anwendungen von strukturierten Sekantenmethoden*, Diploma thesis, Universität Trier, Germany, 1989.

[19] H. J. MARTÍNEZ, *Local and Superlinear Convergence of Structured Secant Methods from the Convex Class*, Ph.D. thesis, Rice Univ., Houston, TX, 1988.

[20] J. MORÉ, B. GARBOW, AND K. HILLSTROM, *Testing unconstrained optimization software*, Trans. Math. Software, 7 (1981), pp. 17–41.

[21] S. OREN, *Self-scaling variable metric algorithms without line-search for unconstrained minimization*, Math. Comp., 27 (1973), pp. 873–885.

[22] R. B. SCHNABEL, *Analyzing and improving quasi-Newton methods for unconstrained optimization*, Ph.D. thesis, Cornell Univ., Ithaca, NY, 1977.

SIAM J. OPTIMIZATION
Vol. 4, No. 1, pp. 130–141, February 1994
© 1994 Society for Industrial and Applied Mathematics
006

EXISTENCE AND DIFFERENTIABILITY OF METRIC PROJECTIONS IN HILBERT SPACES*

ALEXANDER SHAPIRO†

Abstract. This paper considers metric projections onto a closed subset S of a Hilbert space. If the set S is convex, then it is well known that the corresponding metric projections always exist, unique and directionally differentiable at boundary points of S. These properties of metric projections are considered for possibly nonconvex sets S. In particular, existence and directional differentiability of metric projections for certain classes of sets are established and will be referred to as "nearly convex" sets.

Key words. metric projection, Hilbert space, directional differentiability, tangent cones, distance function

AMS subject classifications. 41A50, 41A65, 46C99, 52A30, 52A40

1. Introduction.

Consider a Hilbert space X and let S be a closed subset of X. With the set S are associated the distance function $d_S(x) = \operatorname{dist}(x, S)$,

$$\operatorname{dist}(x, S) = \inf\{\|x - y\| : y \in S\},$$

and the set-valued metric projection

$$\Pi_S(x) = \{y \in S : \|x - y\| = d_S(x)\}.$$

We also consider a selection mapping $P_S(x) \in \Pi_S(x)$, referred to as a metric projection of X onto S. Of course, $P_S(x)$ is defined only at such points x where the set $\Pi_S(x)$ is nonempty.

In cases when the set S is convex there are some well-known properties of P_S. That is, for all $x \in X$ the metric projection $P_S(x)$ exists and is unique, P_S is Lipschitz continous and for every $x \in S$, P_S is directionally differentiable at x (see, e.g., Zarantonello [18]). In this paper we discuss extensions of these results to situations where the set S is not necessarily convex. We also study directional differentiability of P_S at a point $x \notin S$.

Recall that the contingent (Bouligand) cone to S at $x \in S$, denoted by $T_S(x)$ or $T(x, S)$, is formed by vectors y such that there exist $x_n \in S$, $x_n \to x$, and $t_n \to 0^+$ with $t_n^{-1}(x_n - x) \to y$. Its polar (negative dual) cone,

$$N_S(x) = \{y \in X : (y, z) \leq 0 \text{ for all } z \in T_S(x)\}, \tag{1.1}$$

will be referred to as the normal cone to S at x. By $B(x, r)$ we denote the ball

$$B(x, r) = \{y : \|y - x\| \leq r\}.$$

It is said that a mapping $F : X \to Y$, from X into a Banach space Y, is directionally differentiable at x_0 (in the sense of Gâteaux) if the directional derivative

$$F'(x_0, y) = \lim_{t \to 0^+} \frac{F(x_0 + ty) - F(x_0)}{t} \tag{1.2}$$

* Received by the editors February 10, 1992; accepted for publication (in revised form) December 4, 1992.

† School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, Georgia 30332-0205.

exists for all $y \in X$. We also say that F is directionally differentiable at x_0 in the Hadamard sense if

$$F'(x_0, y) = \lim_{\substack{y' \to y \\ t \to 0^+}} \frac{F(x_0 + ty') - F(x_0)}{t}, \tag{1.3}$$

and that F is directionally differentiable in the Fréchet sense if

$$\lim_{y \to 0} \frac{F(x_0 + y) - F(x_0) - F'(x_0, y)}{\|y\|} = 0. \tag{1.4}$$

For a discussion of various relations between these concepts of directional differentiability see, e.g., [4], [9], and [14].

We shall use the following variational principle (see [15] for details). Consider the optimization problems

$$\min_{x \in S} f(x) \tag{1.5}$$

and

$$\min_{x \in T} g(x), \tag{1.6}$$

where $f, g : X \to \Re$ and $S, T \subset X$. Let x_0 be an optimal solution of the problem (1.5), and let $\bar{x}$ be an ϵ-optimal solution of (1.6), i.e., $\bar{x} \in T$ and

$$g(\bar{x}) \leq \inf_{x \in T} g(x) + \epsilon.$$

Suppose that there exist a positive constant α and a neighborhood W of x_0 such that for all $x \in S \cap W$,

$$f(x) \geq f(x_0) + \alpha \|x - x_0\|^2. \tag{1.7}$$

Suppose further that $f(x)$ and $g(x)$ are Lipschitz continuous on W modulus k_1 and k_2, respectively, and that $\bar{x} \in W$. Then

$$\|\bar{x} - x_0\| \leq \alpha^{-1}\kappa + \alpha^{-1/2}\epsilon^{1/2} + 2\delta_1 + \alpha^{-1/2}(k_1\delta_1 + k_2\delta_2)^{1/2}, \tag{1.8}$$

where

$$\delta_1 = \sup_{x \in T \cap W} \operatorname{dist}(x, S \cap W), \tag{1.9}$$

$$\delta_2 = \operatorname{dist}(x_0, T \cap W), \tag{1.10}$$

and κ is a Lipschitz constant of the function $h(x) = g(x) - f(x)$ on W. Note that if $S = T$, then $\delta_1 = \delta_2 = 0$, and if $f \equiv g$, then $\kappa = 0$.

2. Existence of metric projections. In this section we give some sufficient conditions for existence and uniqueness of $P_S(x)$. We consider separately two cases: (i) when points x are sufficiently close to a point $x_0 \in S$, and (ii) when x are near a point $\bar{x}$ such that $P_S(\bar{x})$ exists and is unique. We shall use the following concepts of "nearly convex" sets.

DEFINITION 2.1. We say that the set S is $O(m)$-convex at a point $x_0 \in S$ if there are a neighborhood N of x_0 and a positive constant K such that for all $x, y \in S \cap N$,

$$\operatorname{dist}(y - x, T_S(x)) \leq K\|y - x\|^m. \tag{2.1}$$

DEFINITION 2.2. We say that S is $o(m)$-convex at $x_0 \in S$ if there exist a function $k(x,y)$ and a neighborhood N of x_0 such that for all $x, y \in S \cap N$,

$$\operatorname{dist}(y-x, T_S(x)) \leq k(x,y)\|y-x\|^m \tag{2.2}$$

and

$$\lim_{x,y \to x_0} k(x,y) = 0. \tag{2.3}$$

Remarks. It is not difficult to see that $o(m)$-convexity at a point $x_0 \in S$ implies $O(m)$-convexity at the same point. Also if S is $O(m')$-convex at x_0 and $m' > m$, then $o(m)$-convexity at x_0 follows. (Indeed, take $k(x,y) = K\|y-x\|^{m'-m}$.) When the set S is convex, we have that for all $x, y \in S$, $y-x \in T_S(x)$ and hence $\operatorname{dist}(y-x, T_S(x)) = 0$.

The concept of $o(1)$-convexity was introduced in Shapiro and Al-Khayyal [16] under the name "*near convexity.*" In this paper we shall be dealing mainly with $O(2)$- and $o(2)$-convexity. We shall need the following result. Its proof is similar to the proof of [16, Lem. 1].

LEMMA 2.1. *Suppose that S is $o(m)$-convex at x_0. Then there is a neighborhood N of x_0 such that for all $x_1, x_2 \in S \cap N$ and all $y_1 \in N_S(x_1)$, $y_2 \in N_S(x_2)$,*

$$(y_1 - y_2, x_1 - x_2) \geq -\{k(x_1,x_2)\|y_1\| + k(x_2,x_1)\|y_2\|\}\|x_1 - x_2\|^m. \tag{2.4}$$

Proof. It follows from (2.2) that for any $\epsilon > 0$ there is a point $\bar{x}_2 \in x_1 + T_S(x_1)$ such that

$$\|x_2 - \bar{x}_2\| \leq k(x_1,x_2)\|x_2 - x_1\|^m + \epsilon. \tag{2.5}$$

Since $y_1 \in N_S(x_1)$ and $\epsilon > 0$ is arbitrary we obtain then

$$\begin{aligned}(y_1, x_1 - x_2) &= (y_1, x_1 - \bar{x}_2) + (y_1, \bar{x}_2 - x_2) \geq (y_1, \bar{x}_2 - x_2) \\ &\geq -\|y_1\|\|\bar{x}_2 - x_2\| \geq -k(x_1,x_2)\|y_1\|\|x_1 - x_2\|^m.\end{aligned}$$

Similarly, for $y_2 \in N_S(x_2)$,

$$(y_2, x_2 - x_1) \geq -k(x_2,x_1)\|y_2\|\|x_1 - x_2\|^m.$$

Adding these two inequalities together we obtain (2.4). □

It can be shown in the same way that if S is $O(m)$-convex at x_0, then for all $x_1, x_2 \in S \cap N$ and all $y_1 \in N_S(x_1), y_2 \in N_S(x_2)$,

$$(y_1 - y_2, x_1 - x_2) \geq -2K(\|y_1\| + \|y_2\|)\|x_1 - x_2\|^m. \tag{2.6}$$

We can now formulate the first existence theorem.

THEOREM 2.2. *Suppose that S is $O(2)$-convex at $x_0 \in S$. Then $P_S(x)$ exists and is unique and locally Lipschitzian for all x in a neighborhood of x_0.*

Proof. Consider the neighborhood N specified in Definition 2.1, and let r be a positive number such that $B(x_0, r) \subset N$. For any given point x and $\epsilon > 0$ we can find a point $z \in S$ such that

$$\|x - z\|^2 \leq d_S(x)^2 + \epsilon^2.$$

We then have

$$\|z - x_0\| \leq \|z - x\| + \|x - x_0\| \leq d_S(x) + \epsilon + \|x - x_0\| \leq 2\|x - x_0\| + \epsilon.$$

Moreover, by Ekeland's variational principle [2, p. 255] we can find $\bar{z} \in S$ such that

$$\|x - \bar{z}\|^2 \leq d_S(x)^2 + \epsilon^2, \tag{2.7}$$

$\|z - \bar{z}\| \leq \epsilon$, and $\bar{z}$ is the minimizer of the function

$$g_\epsilon(z) = \|x - z\|^2 + \epsilon\|z - \bar{z}\| \tag{2.8}$$

over S. It follows that

$$\|\bar{z} - x_0\| \leq 2\|x - x_0\| + 2\epsilon. \tag{2.9}$$

We denote by $\Pi(x, \epsilon)$ the set of points $\bar{z} \in S$ satisfying conditions (2.7) and (2.9) and being minimizers over S of the corresponding functions given in (2.8). Clearly, for $\epsilon_1 > \epsilon_2$ we have $\Pi(x, \epsilon_2) \subset \Pi(x, \epsilon_1)$.

Now consider a point x and $\epsilon > 0$ such that $\|x - x_0\| < r/4$ and $\epsilon < r/4$, and let $\bar{z}_1, \bar{z}_2 \in \Pi(x, \epsilon)$. It then follows from (2.9) that $\bar{z}_1, \bar{z}_2 \in S \cap N$. Moreover, by first-order optimality conditions we have that there exist v_1 and v_2 such that $\|v_1\| \leq \epsilon, \|v_2\| \leq \epsilon$, and

$$y_i = x - \bar{z}_i + v_i \in N_S(\bar{z}_i), \qquad i = 1, 2. \tag{2.10}$$

We then obtain from (2.6) that

$$-(y_1 - y_2, \bar{z}_1 - \bar{z}_2) \leq 2K\|\bar{z}_1 - \bar{z}_2\|^2(\|y_1\| + \|y_2\|)$$

and hence

$$\|\bar{z}_1 - \bar{z}_2\|^2 - \|v_1 + v_2\|\|\bar{z}_1 - \bar{z}_2\| \leq 2K\|\bar{z}_1 - \bar{z}_2\|^2(r + 2\epsilon).$$

It follows that

$$\|\bar{z}_1 - \bar{z}_2\|^2 - 2\epsilon\|\bar{z}_1 - \bar{z}_2\| \leq 2K\|\bar{z}_1 - \bar{z}_2\|^2(r + 2\epsilon)$$

and, since $\epsilon < r/4$,

$$(1 - 3rK)\|\bar{z}_1 - \bar{z}_2\| \leq 2\epsilon. \tag{2.11}$$

We further take such $r > 0$ that $r < (3K)^{-1}$ and hence $\alpha = 1 - 3rK > 0$.

Let $\epsilon_n \downarrow 0$ and $\bar{z}_n \in \Pi(x, \epsilon_n)$. It then follows from (2.11) that for $m \geq n$,

$$\|\bar{z}_n - \bar{z}_m\| \leq 2\alpha^{-1}\epsilon_n;$$

hence $\{\bar{z}_n\}$ is a Cauchy sequence. Consequently, $\{\bar{z}_n\}$ converges to a point $z^* \in S$. It also follows from (2.7) that $\|x - z^*\| = d_S(x)$ and hence $z^* = P_S(x)$. Uniqueness of $P_S(x)$ follows now from (2.11) by taking $\epsilon = 0$.

Finally, consider two points x_1, x_2 sufficiently close to x_0 such that $z_1 = P_S(x_1)$ and $z_2 = P_S(x_2)$ do exist. We have that $x_i - z_i \in N_S(z_i)$, $i = 1, 2$, and hence by (2.6),

$$-(x_1 - z_1 - x_2 + z_2, z_1 - z_2) \leq 2K\|z_1 - z_2\|^2(\|x_1 - z_1\| + \|x_2 - z_2\|).$$

Consequently, by taking x_1 and x_2 sufficiently close to x_0 such that

$$\beta = 2K(\|x_1 - z_1\| + \|x_2 - z_2\|) < 1,$$

we obtain

$$\|z_1 - z_2\|^2 - \|x_1 - x_2\|\|z_1 - z_2\| \leq \beta\|z_1 - z_2\|^2$$

and hence

$$\|z_1 - z_2\| \leq (1-\beta)^{-1}\|x_1 - x_2\|.$$

This completes the proof. □

In situations where points x are not necessarily close to S we shall have to impose stronger conditions on "near convexity" of S.

THEOREM 2.3. *For a given point $\bar{x}$, let $x_0 \in \Pi_S(\bar{x})$ and suppose that*

(i) *S is $o(2)$-convex at x_0, and*

(ii) *There is a positive constant α such that*

$$\|\bar{x} - x\|^2 \geq \|\bar{x} - x_0\|^2 + \alpha\|x - x_0\|^2 \tag{2.12}$$

for all $x \in S$.

Then $P_S(x)$ exists, is unique, and is locally Lipschitz continuous for all x in a neighborhood of $\bar{x}$.

Proof. Consider a point x, a positive number ϵ, and let $z^* \in S$ be such that

$$\|x - z^*\|^2 \leq d_S(x)^2 + \epsilon^2.$$

That is, z^* is an ϵ^2-optimal solution of the problem of minimization of $\|x-z\|^2$ subject to $z \in S$. Note that condition (2.12) corresponds to the second-order growth condition (1.7) for $f(z) = \|\bar{x} - z\|^2$. It then follows from (1.8) that

$$\|z^* - x_0\| \leq \alpha^{-1}\kappa + \alpha^{-1/2}\epsilon,$$

where $\kappa = 2\|x - \bar{x}\|$. Furthermore, by Ekeland's variational principle we can then find a point $\bar{z} \in S$ such that $\|z^* - \bar{z}\| \leq \epsilon$, that inequality (2.7) holds, and that $\bar{z}$ is the minimizer of the corresponding function given in (2.8). Proceeding now as in the proof of Theorem 2.2 and using (2.4) (instead of (2.6)), we obtain vectors v_1, v_2, such that $\|v_1\| \leq \epsilon$, $\|v_2\| \leq \epsilon$, and

$$\|\bar{z}_1 - \bar{z}_2\|^2 - \|v_1 + v_2\|\|\bar{z}_1 - \bar{z}_2\| \leq \{k(\bar{z}_1, \bar{z}_2)\|y_1\| + k(\bar{z}_2, \bar{z}_1)\|y_2\|\}\|\bar{z}_1 - \bar{z}_2\|^2,$$

where $y_i = x - \bar{z}_i + v_i$. Now for x sufficiently close to $\bar{x}$ and ϵ sufficiently small we can choose $\bar{z}_1, \bar{z}_2$ sufficiently close to x_0 such that

$$\beta = k(\bar{z}_1, \bar{z}_2)\|y_1\| + k(\bar{z}_2, \bar{z}_1)\|y_2\| < 1.$$

Then

$$\|\bar{z}_1 - \bar{z}_2\| \leq 2(1-\beta)^{-1}\epsilon,$$

and we can complete the proof proceeding as in the proof of Theorem 2.2. □

Consider now a situation where the set S is given in the form

$$S = \{x \in X : g(x) \in K\}, \tag{2.13}$$

with $g(x)$ being a continuously differentiable mapping from X into a Banach space Y and K being a closed convex cone in Y. Consider a point $x_0 \in S$ regular in the sense of Robinson [10], that is,

$$0 \in \text{ int } \{g(x_0) + Dg(x_0)X - K\}. \tag{2.14}$$

It then follows from the Robinson–Ursescu [11], [17] stability theorem that there is a neighborhood N of x_0 and a constant $c > 0$ such that for all $x, y \in S \cap N$,

$$\operatorname{dist}(y - x, T_S(x)) \leq c\|g(y) - g(x) - Dg(x)(y - x)\| \tag{2.15}$$

(see the proof of Theorem 2 in [16]). By the Mean Value Theorem we obtain from (2.15) that if, in addition, $Dg(x)$ is Lipschitz continuous near x_0 (in the operator norm), then S is $O(2)$-convex at x_0. Moreover, if $g(x)$ is twice continuously differentiable and $D^2_{xx}g(x_0) = 0$, then $o(2)$-convexity follows.

Note that here the contingent cone $T_S(x)$ is given (for all points x sufficiently close to x_0) by

$$T_S(x) = \{y : Dg(x)y \in T_K(g(x))\} \tag{2.16}$$

and hence is *convex.*

3. Directional differentiability of metric projections at boundary points. It is known that if the set S is convex and $x_0 \in S$, then for any $y \in X$ and $t \geq 0$,

$$P_S(x_0 + ty) = x_0 + tP_{T(x_0,S)}(y) + o(t) \tag{3.1}$$

(see [18]). That is, P_S is directionally differentiable at x_0 (in the sense of Gâteaux) and

$$P'_S(x_0; y) = P_{T(x_0,S)}(y). \tag{3.2}$$

Moreover, since for any $x_1, x_2 \in X$,

$$\|P_S(x_1) - P_S(x_2)\| \leq \|x_1 - x_2\|,$$

i.e., P_S is Lipschitz continuous modulus one, this implies that P_S is directionally differentiable at x_0 in the Hadamard sense. We extend this result to nonconvex sets as follows.

THEOREM 3.1. *Suppose that $x_0 \in S$ and that*

(i) *The contingent cone $T_S(x_0)$ is convex*

(ii)

$$\lim_{\substack{x \to x_0 \\ x \in S}} \frac{\operatorname{dist}(x - x_0, T_S(x_0))}{\|x - x_0\|} = 0, \tag{3.3}$$

(iii) *For every $v \in T_S(x_0)$,*

$$\lim_{t \to 0^+} \frac{\operatorname{dist}(x_0 + vt, S)}{t} = 0. \tag{3.4}$$

Then for any fixed $y \in X$ and for $P_S(x_0 + ty) \in \Pi_S(x_0 + ty)$ with $t \geq 0$,

$$P_S(x_0 + ty) = x_0 + tP_{T(x_0,S)}(y) + o(t). \tag{3.5}$$

Proof. Without loss of generality we can assume that $x_0 = 0$. For a given vector y and $t > 0$ consider the optimization problems

$$\min_{x \in T(0,S)} f(x, ty) \tag{3.6}$$

and

$$\min_{x \in S} f(x, ty), \tag{3.7}$$

where $f(x_1, x_2) = \|x_1 - x_2\|^2$. Note that the optimal solution of (3.6) is $x^* = P_{T(0,S)}(ty)$, which always exists since $T_S(0)$ is convex (and closed). An optimal solution of (3.7) (if it exists) is $\bar{x} = P_S(ty)$. We have here that for any $x \in T_S(0)$,

$$f(x, ty) \geq f(x^*, ty) + \|x - x^*\|^2, \tag{3.8}$$

i.e., the second-order growth condition for (3.6) holds. Indeed,

$$\|x - ty\|^2 - \|x^* - ty\|^2 = 2(ty - x^*, x^* - x) + \|x^* - x\|^2$$

and, since $ty - x^* \in N_{T(0,S)}(x^*)$ and $T_S(0)$ is convex,

$$(ty - x^*, x^* - x) \geq 0.$$

It follows then from (1.8) that

$$\|\bar{x} - x^*\| \leq 2\delta_1 + (k\delta_1 + k\delta_2)^{1/2}, \tag{3.9}$$

where

$$\delta_1 = \delta_1(t) = \sup\{\operatorname{dist}(x, T_S(0)) : x \in S \cap B(0, r)\},$$
$$\delta_2 = \delta_2(t) = \operatorname{dist}(x^*, S),$$

$r = 2t\|y\|$, and k is the Lipschitz constant of $f(\cdot, ty)$ on the ball $B(0, r)$. Note that $x^*, \bar{x} \in B(0, r)$ and that

$$k \leq 2(t\|y\| + r) = O(t).$$

Now it follows from condition (ii) that $\delta_1(t) = o(r) = o(t)$ and from (iii), for $v = P_{T(0,S)}(y) \in T_S(0)$, that

$$\delta_2(t) = \operatorname{dist}(tv, S) = o(t). \tag{3.10}$$

It follows then from (3.9) that

$$\|P_S(ty) - P_{T(0,S)}(ty)\| = \|\bar{x} - x^*\| = o(t),$$

and hence the proof is complete. □

Remarks. If $P_S(x)$ exists for all x in a neighborhood of x_0 (for example, if S is $O(2)$-convex at x_0), then formula (3.5) implies that P_S is directionally differentiable at x_0 and the directional derivative $P'_S(x_0, y)$ is given by $P_{T(x_0,S)}(y)$. Moreover, since $T(x_0, S)$ was supposed to be convex, we have that for any y and y',

$$\|P_{T(x_0,S)}(y') - P_{T(x_0,S)}(y)\| \leq \|y' - y\|.$$

Then, by replacing $v = P_{T(x_0,S)}(y)$ with $v' = P_{T(x_0,S)}(y')$ in (3.10) and taking $y' \to y$, we obtain that assumptions (i)–(iii) of Theorem 3.1 actually imply Hadamard directional differentiability of P_S at x_0.

If the set S is defined in the form (2.13) and regularity condition (2.14) holds, then conditions (i)–(iii) of Theorem 3.1 follow by the Robinson–Ursescu stability theorem. In this case we obtain that $P_S(x)$ exists for all x near $x_0 \in S$ and is Hadamard directionally differentiable at x_0.

In order to ensure directional differentiability of P_S, at $x_0 \in S$, in the sense of Fréchet we will have to replace condition (iii) by

$$\lim_{\substack{y \to 0 \\ y \in T(x_0, S)}} \frac{\operatorname{dist}(x_0 + y, S)}{\|y\|} = 0. \tag{3.11}$$

This condition is considerably stronger than the corresponding condition (3.4) and, in infinite-dimensional spaces, does not necessarily hold even for convex sets S. (In finite-dimensional spaces conditions (3.4) and (3.11) are equivalent.) Cones $T_S(x_0)$ satisfying (3.3) and (3.11) were called the *approximating cones* in [12].

4. Directional differentiability of metric projections at nonboundary points. In cases when $\bar{x}$ is not contained in S, the metric projection P_S can be directionally nondifferentiable at $\bar{x}$ even if the set S is convex. An example of such a convex set S in $\Re^3$ is given in Kruskal [8]. We show that directional differentiability of P_S follows if the set S is sufficiently "flat" at $x_0 = P_S(\bar{x})$. With a point $\bar{x}$ and $x_0 = P_S(\bar{x})$ we associate the cone

$$C(\bar{x}) = \{v \in T_S(x_0) : (v, \bar{x} - x_0) = 0\}. \tag{4.1}$$

THEOREM 4.1. *For a given point $\bar{x}$, let $x_0 \in \Pi_S(\bar{x})$ and suppose that the following conditions hold:*

(i) *The contingent cone $T_S(x_0)$ is convex.*

(ii) *There is a positive constant α such that*

$$\|\bar{x} - x\|^2 \geq \|\bar{x} - x_0\|^2 + \alpha \|x - x_0\|^2 \tag{4.2}$$

for all $x \in S$.

(iii)

$$\lim_{\substack{x \to x_0 \\ x \in S}} \frac{\operatorname{dist}(x - x_0, T_S(x_0))}{\|x - x_0\|^2} = 0. \tag{4.3}$$

(iv) *For any curve $v(t) \in T_S(x_0)$, $t \geq 0$, such that $v(t) = t\bar{v} + o(t)$ and $\bar{v} \in C(\bar{x})$,*

$$\lim_{t \to 0+} \frac{\operatorname{dist}(x_0 + v(t), S)}{t^2} = 0. \tag{4.4}$$

Then for any fixed $y \in X$ and for $P_S(\bar{x} + ty) \in \Pi_S(\bar{x} + ty)$ with $t \geq 0$,

$$P_S(\bar{x} + ty) = x_0 + tP_{C(\bar{x})}(y) + o(t). \tag{4.5}$$

In order to prove Theorem 4.1 we shall use the results of the following lemmas, which may also be of independent interest.

LEMMA 4.2. *Let Σ be a closed convex set such that $P_\Sigma(\bar{x}) = x_0$ and*

$$\lim_{\substack{x \to x_0 \\ x \in S}} \frac{\operatorname{dist}(x, \Sigma)}{\|x - x_0\|^2} = 0, \tag{4.6}$$

$$\lim_{t\to 0+} \frac{\operatorname{dist}(P_\Sigma(\bar{x}+ty), S)}{t^2} = 0, \tag{4.7}$$

and suppose that condition (ii) *of Theorem* 4.1 *holds.*

Then

$$P_S(\bar{x}+ty) = P_\Sigma(\bar{x}+ty) + o(t), \qquad t \geq 0. \tag{4.8}$$

Proof. Let us first observe that, by (1.8), it follows from condition (4.2) that

$$\|P_S(x) - P_S(\bar{x})\| \leq 2\alpha^{-1}\|x - \bar{x}\|. \tag{4.9}$$

Consider the following optimization problems:

$$\min_{x\in\Sigma} f(x, \bar{x}+ty) \tag{4.10}$$

and

$$\min_{x\in S} f(x, \bar{x}+ty), \tag{4.11}$$

where $f(x_1, x_2) = \|x_1 - x_2\|^2$. Let $x^* = P_\Sigma(\bar{x}+ty)$ and $x' = P_S(\bar{x}+ty)$, i.e., x^* and x' are optimal solutions of problems (4.10) and (4.11), respectively. Note that it follows from (4.9) that $\|x' - x_0\| \leq \alpha^{-1}t\|y\|$, and because of the convexity of Σ, $\|x^* - x_0\| \leq t\|y\|$. Therefore $x', x^* \in B(x_0, r)$, where $r = \max\{2\alpha^{-1}t\|y\|, t\|y\|\}$.

Since Σ is convex we have that

$$f(x, \bar{x}+ty) \geq f(x^*, \bar{x}+ty) + \|x - x^*\|^2 \tag{4.12}$$

for all $x \in \Sigma$. It follows then by (1.8) that

$$\|x' - x^*\| \leq 2\delta_1 + (k\delta_1 + k\delta_2)^{1/2}, \tag{4.13}$$

where

$$\delta_1 = \delta_1(t) = \sup\{\operatorname{dist}(x, \Sigma) : x \in S \cap B(x_0, r)\},$$
$$\delta_2 = \delta_2(t) = \operatorname{dist}(x^*, S),$$

and k is the Lipschitz constant of $f(\cdot\,, \bar{x}+ty)$ on the ball $B(x_0, r)$. Moreover, we have that

$$k \leq 2(r + \|x_0 - \bar{x}\| + t\|y\|)$$

and that $r = O(t)$. It follows from (4.6) that $\delta_1 = o(r^2)$ and hence $\delta_1 = o(t^2)$, and from (4.7) that $\delta_2 = o(t^2)$. We obtain then from (4.13) that

$$\|x' - x^*\| = o(t)$$

and hence the proof is complete. □

LEMMA 4.3. *Let K be a convex, closed cone, $K^- = N_K(0)$ be its negative dual, $v \in K^-$, and consider the cone*

$$K_0 = \{x \in K : (x, v) = 0\}.$$

Then for a given $y \in X$ and $t \geq 0$,

$$P_K(v + ty) = tP_{K_0}(y) + o(t). \tag{4.14}$$

Proof. We can write (see, e.g., [18])

$$P_K(v + ty) = v + ty - P_{K^-}(v + ty). \tag{4.15}$$

Also since K^- is convex and $v \in K^-$, we have that P_{K^-} is directionally differentiable at v and

$$P_{K^-}(v + ty) = v + tP_{T(v,K^-)}(y) + o(t). \tag{4.16}$$

Note that the contingent cone $T(v, K^-)$ is given by the topological closure of $K^- + [v]$, where $[v]$ denotes the linear space generated by v, and hence the polar (negative dual) of $T(v, K^-)$ is the cone K_0. Therefore,

$$P_{T(v,K^-)}(y) = y - P_{K_0}(y). \tag{4.17}$$

Equations (4.15)–(4.17) imply (4.14).

Proof of Theorem 4.1. Consider $\Sigma = x_0 + T_S(x_0)$. Since $\bar{x} - x_0 \in N_S(x_0)$ we have that $x_0 = P_\Sigma(\bar{x})$. Also, by Lemma 4.3,

$$P_\Sigma(\bar{x} + ty) = x_0 + tP_{C(\bar{x})}(y) + o(t). \tag{4.18}$$

In order to complete the proof we can now apply the result (4.8) of Lemma 4.2. Note that here condition (4.6) follows from (4.3) and condition (4.7) follows from (4.4) because of (4.18). □

If the set S is convex, conditions (i)–(iii) of Theorem 4.1 are automatically satisfied. In this case, condition (iv) implies that P_S is directionally differentiable at x_0, in the sense of Hadamard, and that

$$P'_S(\bar{x}, y) = P_{C(\bar{x})}(y). \tag{4.19}$$

It is known that in general differentiability properties of P_S are influenced by second-order behavior of the boundary of S (cf. [1], [5], [7], and [13]). In this respect condition (iv) ensures that S is sufficiently "directionally flat" at the point $x_0 = P_S(\bar{x})$ and consequently only first-order tangential properties of S at x_0 are reflected in formula (4.19).

Condition (iv) follows from and, in finite-dimensional spaces with $\bar{v} \in C(\bar{x})$ relaxed to $\bar{v} \in T_S(x_0)$, is equivalent to

$$\lim_{\substack{x \to x_0 \\ x \in \Sigma}} \frac{\operatorname{dist}(x, S)}{\|x - x_0\|^2} = 0, \tag{4.20}$$

where $\Sigma = x_0 + T_S(x_0)$. Sets S and Σ satisfying (4.6) and (4.20) were called 2-tangent at x_0, in Auslender and Cominetti [3].

Finally we compare Theorem 4.1 with corresponding results of Haraux [6]. It was shown in [6] that P_S is directionally differentiable and formula (4.19) holds for a class of convex sets S that were called *polyhedric*. In finite-dimensional spaces polyhedric convex sets in the sense of Haraux coincide with ordinary polyhedral convex sets. Clearly a set S does not need to be polyhedral in order to satisfy (4.20). We also give

now an example of a convex, polyhedric set S that does not satisfy condition (iv). Therefore, the results of Theorem 4.1 and those of Haraux do not imply each other.

Consider the Hilbert space $X = L^2[-1, 1]$. Then the set

$$S = \{x(t) \in X : x(t) \geq 0 \text{ for all } t \in [-1, 1]\}$$

forms a convex, polyhedric set in X. Consider also the following points (functions) in X, $\bar{x}(t) = -1$ for $-1 \leq t \leq 0$, $\bar{x}(t) = 1$ for $0 < t \leq 1$, and $\bar{v}(t) = 0$ for $-1 \leq t \leq 0$, $\bar{v}(t) = -t^{-1/3}$ for $0 < t \leq 1$. It is then not difficult to see that the projection $x_0 = P_S(\bar{x})$ is given by $x_0(t) = 0$ for $-1 \leq t \leq 0$ and $x_0(t) = 1$ for $0 < t \leq 1$, and that

$$T_S(x_0) = \{x(t) \in X : x(t) \geq 0 \text{ for all } t \in [-1, 0]\}.$$

Therefore, $\bar{v} \in T_S(x_0)$ and $(\bar{v}, \bar{x} - x_0) = 0$, i.e., $\bar{v} \in C(\bar{x})$. Moreover, for $0 < \tau \leq 1$,

$$P_S(x_0 + \tau\bar{v})(t) = \begin{cases} 0, & -1 \leq t \leq \tau^3, \\ 1 - \tau t^{-1/3}, & \tau^3 \leq t \leq 1, \end{cases}$$

and

$$\operatorname{dist}(x_0 + \tau\bar{v}, S), = \left[\int_0^{\tau^3} (1 - \tau t^{-1/3})^2 dt\right]^{1/2} = \tau^{3/2}.$$

Consequently, condition (4.4) does not hold here.

Haraux's result is an extension of the following construction. Let $X = X_1 \times X_2$ be the product of two Hilbert spaces X_1 and X_2, and let $S = S_1 \times S_2$, with S_1 being a closed convex subset of X_1 and S_2 being a closed convex cone in X_2. Then it follows from Lemma 4.3 that P_S is directionally differentiable at any point $\bar{x} = (\bar{x}_1, \bar{x}_2)$ such that $\bar{x}_1 \in S_1$ and $\bar{x}_2 \in S_2^-$. Clearly, condition (4.4) does not need to hold here at the corresponding point $x_0 = (\bar{x}_1, 0)$.

REFERENCES

[1] T. J. Abatzoglou, *The minimum norm projection on C^2 manifolds in R^n*, Trans. Amer. Math. Soc., 243 (1978), pp. 115-122.

[2] J. P. Aubin and I. Ekeland, *Applied Nonlinear Analysis*, Wiley, New York, 1984.

[3] A. Auslender and R. Cominetti, *A comparative study of multifunction differentiability with applications in mathematical programming*, Math. Oper. Res., 16 (1991), pp. 240–258.

[4] V. I. Averbukh and O. G. Smolyanov, *The various definitions of the derivative in linear topological spaces*, Russian Mathematical Surveys, 23 (1968), pp. 67–113.

[5] S. Fitzpatrick and R. R. Phelps, *Differentiability of the metric projection in Hilbert space*, Trans. Amer. Math. Soc., 270 (1982), pp. 483–501.

[6] A. Haraux, *How to differentiate the projection on a convex set in Hilbert space. Some applications to variational inequalities*, J. Math. Soc. Japan, 29 (1977), pp. 615–631.

[7] R. B. Holmes, *Smoothness of certain metric projections on Hilbert space*, Trans. Amer. Math. Soc., 184 (1973), pp. 87–100.

[8] J. Kruskal, *Two convex counterexamples: A discontinuous envelope function and a non-differentiable nearest point mapping*, Proc. Amer. Math. Soc., 23 (1969), pp. 697–703.

[9] M. Z. Nashed, *Differentiability and related properties of nonlinear operators: Some aspects of the role of differentials in nonlinear functional analysis*, Nonlinear Functional Analysis and Applications, L. B. Rall, ed., Academic Press, New York, 1971, pp. 103–309.

[10] S. M. Robinson, *Stability theorems for systems of inequalities, Part* II: *Differentiable nonlinear systems*, SIAM J. Numer. Anal., 13 (1976), pp. 497–513.

[11] ———, *Regularity and stability for convex multivalued functions*, Math. Oper. Res., 1 (1976), pp. 130–143.

[12] A. SHAPIRO, *On differentiability of metric projections in R^n, 1: Boundary case*, Proc. Amer. Math. Soc., 99 (1987), pp. 123–128.

[13] ———, *Sensitivity analysis of nonlinear programs and differentiability properties of metric projections*, SIAM J. Control and Optimization, 26 (1988), pp. 628–645.

[14] ———, *On concepts of directional differentiability*, J. Optim. Theory Appl., 66 (1990), pp. 477–487.

[15] ———, *Perturbation analysis of optimization problems in Banach spaces*, Numer. Funct. Anal. Optim., 13 (1992), pp. 97–116.

[16] A. SHAPIRO AND F. AL-KHAYYAL, *First order conditions for isolated locally optimal solutions*, J. Optim. Theory Appl., 77 (1993), pp. 189–196.

[17] C. URSESCU, *Multifunctions with convex closed graph*, Czechoslovak Math. J., 25 (1975), pp. 438–441.

[18] E. H. ZARANTONELLO, *Projections on convex sets in Hilbert space and spectral theory*, in contributions to *Nonlinear Functional Analysis*, Academic Press, New York, 1971, pp. 237–424.

SIAM J. OPTIMIZATION
Vol. 4, No. 1, pp. 142–145, February 1994
© 1994 Society for Industrial and Applied Mathematics
007

APPROXIMATING ORACLE MACHINES FOR COMBINATORIAL OPTIMIZATION*

SHMUEL ONN†

Abstract. For every k, an oracle Turing machine M_k and rational polytopes $P_k(S)$ for all n and $S \subseteq \{0,1\}^n$ are constructed; by querying from the set S given as an oracle, M_k^S solves the separation problem over $P_k(S)$ in strongly polynomial time, performing $O(n^{3k})$ arithmetic operations. Each of the polytopes $P_k(S)$ approximates S in the sense $P_k(S) \cap \{0,1\}^n = S$ and $P_k(S) \subseteq [0,1]^n$, and for all k, $P_{k+1}(S)$ is contained in $P_k(S)$. As a result, other oracle machines are obtained that, querying from the oracle S, maximize in polynomial time linear functionals over approximations for S obtained from $P_k(S)$ by applying the cones-of-matrices cutting operator of Lovász and Schrijver a constant (possibly zero) number of times. Thus, this construction enables a systematic application of the cones-of-matrices scheme to any combinatorial optimization problem.

Key words. combinatorial optimization

AMS subject classifications. 52B55, 68Q25, 90C27

1. Introduction.

Combinatorial optimization involves the problem of maximizing a rational linear functional $x \in \mathbb{Q}^n$ over a given set $S \subseteq \{0,1\}^n$. Typically, S is not listed explicitly, and the natural input size for the problem is $n + b(x)$, where $b(x)$ is the number of bits encoding x. A setting which makes it possible to let the input consist of (n,x) only is that of an oracle Turing machine (see [2]). We would like such a machine N, equipped with the set of interest $S \subseteq \{0,1\}^n$ as an oracle, denoted N^S, to maximize x over S. While time complexity $O(2^n b(x))$ is easily obtained by an exhaustive search of S, it is unlikely that such a machine exists that runs in time bounded by a polynomial in $n + b(x)$ that is independent of the oracle being queried.

A fruitful approach, taken in much of the literature on combinatorial optimization, is to restrict the set S to come from a specified class, say restrict S to be the set of characteristic vectors of stable sets in some bipartite graph. However, even in this setting, efficient machines are often unlikely to exist. Motivated by the recent work of Lovász and Schrijver [4], we take here an alternative approach, where the set S can be arbitrary, but N^S is allowed to maximize over a certain approximation of S rather than S. Appealing to the embedding of $S \subseteq \{0,1\}^n \subset \mathbb{Q}^n$ in rational affine space, an *approximation* of S is a rational convex polytope P that is contained in the unit cube $[0,1]^n$ and satisfies $P \cap \{0,1\}^n = S$. In [4], an operator $\mathcal{N}$ was introduced that produces, from a given approximation P of $S \subseteq \{0,1\}^n$, another one, $\mathcal{N}(P)$, with the property $P = \mathcal{N}^0(P) \supseteq \mathcal{N}(P) \supseteq \mathcal{N}^2(P) \supseteq \cdots \supseteq \mathcal{N}^n(P) = \text{conv}(S)$.

Furthermore, an oracle machine was shown to exist that, querying from a separation oracle for P (see [3]), maximizes linear functionals over $\mathcal{N}(P)$ in polynomial time. This naturally raises the following question. Does there exist approximations $P(S)$ for all n and $S \subseteq \{0,1\}^n$, and a single oracle machine M, such that M^S solves the separation problem (see below) over $P(S)$ in polynomial time for all S? In the following section, we provide a positive answer by constructing a family of such approximations and machines. As a result, other oracle machines are obtained that, querying from the

* Received by the editors March 23, 1992; accepted for publication (in revised form) December 3, 1992.

† Center for Discrete Mathematics and Theoretical Computer Science, Rutgers University, P.O. Box 1179, Piscataway, New Jersey 08855-1179. This work was completed while the author was at the School of Operations Research and Industrial Engineering, Cornell University, Ithaca, New York, and was supported in part by the Mathematical Sciences Institute of Cornell University.

oracle S, maximize in polynomial time linear functionals over approximations for S obtained from our oracle machine by applying the operator $\mathcal{N}$ of [4] a constant (possibly zero) number of times. Thus, our construction enables a systematic application of the scheme in [4] to any combinatorial optimization problem.

2. Approximating machines. In [1] (see [3]) a certain algorithmic equivalence was established between the problem of maximizing linear functionals over a rational polytope P and the following *separation* problem over P: given $x \in \mathbb{Q}^n$, decide whether $x \in P$ and, if not, provide a vector $y \in \mathbb{Q}^n$ separating x from P, i.e., satisfying $\langle y, z\rangle < \langle y, x\rangle$ for all $z \in P$. We shall start by studying separation, rather than maximization, over our approximations. An oracle machine M will be called *approximating* if M^S solves the separation problem over some approximation $P(S)$ of S for every n and every $S \subseteq \{0,1\}^n$. All our machines will be designed to accept input of the form (n, x) (where $x \in \mathbb{Q}^n$) only. We shall say that M is a *polynomial time* oracle machine if the running time of M^S is bounded by a polynomial (in the size $n + b(x)$ of the input) which is independent of the oracle S being queried. We shall say that M is *strongly polynomial time* (see [3] and [5]) if it is polynomial time and the number of arithmetic operations that M^S performs is bounded by a polynomial in n only (independent of x and the oracle S).

For every positive integer k, we construct an approximation $P_k(S)$ of S for every S, and a related strongly polynomial time approximating oracle machine M_k. Our approximations satisfy $P_1(S) \supseteq P_2(S) \cdots \supseteq P_n(S) \supseteq \mathrm{conv}(S)$, and $P_n(S)$ is the smallest intersection of $[0,1]^n$ and a polytope with facet normals in $\{-1,1\}^n$ containing S.

We need some notation. Throughout the rest of the note, a, b, c denote elements (or variables) of the unit cube $[0,1]^n$, u, v, w denote elements of $\{0,1\}^n$, x, y, z denote elements of $\mathbb{Q}^n$; and n, m, k denote positive integers. The inner product of two vectors x, y will be abbreviated $\langle x, y\rangle = xy$. We let $e = (1, \ldots, 1)$, and, for any x, we let $\bar{x} = 2x - e$ and $|x| = ||x||_1 = \sum_{i=1}^n |x_i|$. Finally, for any v and m (to be interpreted as "for any $v \in \{0,1\}^n$ and any $m \in \mathbb{N}$") we let $B(v,m) = \{u : \ |u - v| < m\}$.

Now, let k, n, and $S \subseteq \{0,1\}^n$ be arbitrary. For any v, let

$$k_v = \min\{\inf_{u \in S} |u - v|, \ k\}.$$

Define

$$P_k(S) = \cap_v \{a : \ |a - v| \geq k_v\}.$$

It is easy to check that

$$S = \cap_v \{u : \ |u - v| \geq k_v\},$$

so that $P_k(S) \cap \{0,1\}^n = S$. The observation below is easy to verify (but does not hold in general for rational vectors outside the unit cube).

Observation. For every a and v we have $|a - v| = |a| + |v| - 2av = |v| - \bar{v}a$.

It follows that $P_k(S) = \cap_v \{x : \ \bar{v}x \leq |v| - k_v\} \cap [0,1]^n$ is a convex polytope, so $P_k(S)$ is indeed an approximation of S.

For any a let $[a]$ be the *rounding* of a, i.e., the vector having $[a]_i = 0$ if $a_i < \frac{1}{2}$ and $[a]_i = 1$ if $a_i \geq \frac{1}{2}$. We have the following proposition.

PROPOSITION 2.1. *A point a does not belong to $P_k(S)$ if and only if there exists $v \in B([a], 2k)$ for which $|a - v| < k_v$.*

Proof. By definition of $P_k(S)$, if such v exists then $a \notin P_k(S)$. To prove the converse, note first that for every a and v we have $|a - [a]| \leq |a - v|$. Now, if $a \notin P_k(S)$,

then there exists some v for which $|a-v| < k_v$, but then $|a-[a]| \leq |a-v| < k_v$ as well; so by the triangle inequality $|v-[a]| \leq |v-a| + |a-[a]| < 2k_v \leq 2k$, and $v \in B([a], 2k)$. □

THEOREM 2.2. *For every k, there exists a strongly polynomial time oracle machine M_k such that, for all n and $S \subseteq \{0,1\}^n$, M_k^S solves the separation problem over $P_k(S)$.*

Proof. Given input (n, x), it is easy to check whether $x \in [0,1]^n$, and if not, provide a vector separating x from $[0,1]^n$, hence from $P_k(S)$. So, we may assume that $x = a \in [0,1]^n$, and obtain $[a]$. By the proposition above, $a \in P_k(S)$ unless there exists $v \in B([a], 2k)$ for which $|a-v| < k_v$, in which case the vector $\bar{v}$ separates a from $P_k(S)$, since $\bar{v}y \leq |v| - k_v < \bar{v}a$ for all $y \in P_k(S)$. Thus, we have to compute k_v and check if $|a-v| < k_v$ for no more than $|B([a], 2k)| = \sum_{i=0}^{2k-1} \binom{n}{i} < n^{2k}$ elements v. Now, clearly,

$$k_v = \min\{\inf\{|u-v| : \ u \in S \cap B(v,k)\}, \ k\};$$

so for any v we can compute k_v by querying whether $u \in S$ for no more than $|B(v,k)| < n^k$ elements u, and doing $O(n^k)$ comparisons. Thus, the overall time required for carrying out this procedure is polynomial in $n + b(x)$, and the number of arithmetic operations is of order n^{3k}. □

Using the terminology of [3], the *facet complexity* of a rational polytope is the maximum number of bits needed to encode any of its facets. It is easy to see that for all k, our $P_k(S)$ has facet complexity of order n. Thus, by [3, Thm. 2.3.3], our machine M_k can be converted into a polynomial time oracle machine $N_{k,0}$ that will maximize linear functionals over our approximations. Turning to the scheme of [4], it follows from the definition of the operator $\mathcal{N}$ that, for every fixed m, the facet complexity of $\mathcal{N}^m(P_k(S))$ is bounded by a polynomial in n. Also, by the results of [4], our polynomial time oracle machine M_k separating over $P_k(S)$ can be used to obtain a polynomial time oracle machine separating over $\mathcal{N}^m(P_k(S))$. Applying [3, Thm. 2.3.3] again, we obtain the following conclusion.

COROLLARY 2.3. *For every k and m, a polynomial time oracle machine $N_{k,m}$ exists such that, for all n and $S \subseteq \{0,1\}^n$, $N_{k,m}^S$ maximizes rational linear functionals over $\mathcal{N}^m(P_k(S))$, providing an approximate solution to the maximization problem over S. When $n \leq m$, $N_{k,m}^S$ solves the maximization problem over S itself.*

3. Discussion. As noted before, for all n and $S \subseteq \{0,1\}^n$, our $P_n(S)$ is the smallest intersection of $[0,1]^n$ and a polytope with facet normals in $\{-1,1\}^n$ containing S. It is an interesting question whether another family of approximations $Q_k(S)$ could be constructed, with algorithmic properties similar to our $P_k(S)$, and the stronger geometric property that $Q_n(S) \subseteq P_n(S)$ be the smallest polytope with facet normals in $\{-1,0,1\}^n$ containing S.

It is easy to see that when $n > 2k$, the polytope $P_k(S)$ is full-dimensional for all $S \subseteq \{0,1\}^n$. If $\mathrm{conv}(S)$ is not full-dimensional, it is desirable to use our approximations intersected with the affine hull of S which is often known, that is, to modify our machines M_k so that M_k^S will solve the separation problem over $P_k(S) \cap \mathrm{aff}(S)$ rather than $P_k(S)$. In particular, if a language $L \subseteq \cup_{n \geq 0}\{0,1\}^n$ admits a polynomial time algorithm that solves the separation problem over $\mathrm{aff}(L_n)$ for all n (where $L_n = L \cap \{0,1\}^n$), then indeed, both M_k and $N_{k,m}$ can be modified so that in polynomial time $M_k^{L_n}$ will solve the separation problem over $P_k(L_n) \cap \mathrm{aff}(L_n)$, and $N_{k,m}^{L_n}$ will maximize over $\mathcal{N}^m(P_k(L_n) \cap \mathrm{aff}(L_n))$.

A typical example is provided by the language of characteristic vectors of Hamiltonian cycles in complete graphs, where a polynomial time separation algorithm for the affine hull exists.

Acknowledgments. The author thanks Robert G. Bland and Éva Tardos for conversations about this note, and Éva Tardos and the referees for helpful comments on an earlier draft.

REFERENCES

[1] M. GRÖTSCHEL, L. LOVÁSZ, AND A. SCHRIJVER, *The ellipsoid method and its consequences in combinatorial optimization*, Combinatorica, 1 (1981), pp. 169–197.

[2] J. E. HOPCROFT AND J. D. ULLMAN, *Introduction to Automata Theory, Languages, and Computation*, Addison-Wesley, Reading, MA, 1979.

[3] L. LOVÁSZ, *An Algorithmic Theory of Numbers, Graphs and Convexity*, CBMS-NSF Regional Conference Series on Applied Mathematics, 50, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1986.

[4] L. LOVÁSZ AND A. SCHRIJVER, *Cones of matrices and set-functions and* 0–1 *optimization*, SIAM J. Optimization, 1 (1991), pp. 166–190.

[5] E. TARDOS, *A strongly polynomial algorithm to solve combinatorial linear programs*, Oper. Res., 34 (1986), pp. 250–256.

SIAM J. OPTIMIZATION
Vol. 4, No. 1, pp. 146–158, February 1994
© 1994 Society for Industrial and Applied Mathematics
008

STRONG ROTUNDITY AND OPTIMIZATION*

J. M. BORWEIN† AND A. S. LEWIS†

Abstract. Standard techniques from the study of well-posedness show that if a fixed convex objective function is minimized in turn over a sequence of convex feasible regions converging Mosco to a limiting feasible region, then the optimal solutions converge in norm to the optimal solution of the limiting problem. Certain conditions on the objective function are needed as is a constraint qualification. If, as may easily occur in practice, the constraint qualification fails, stronger set convergence is required, together with stronger analytic/geometric properties of the objective function: strict convexity (to ensure uniqueness), weakly compact level sets (to ensure existence and weak convergence), and the Kadec property (to deduce norm convergence). By analogy with the L_p norms, such properties are termed "strong rotundity." A very simple characterization of strongly rotund integral functionals on L_1 is presented that shows, for example, that the Boltzmann–Shannon entropy $\int x \log x$ is strongly rotund. Examples are discussed, and the existence of everywhere- and densely-defined strongly rotund functions is investigated.

Key words. strongly rotund, well-posed, set-convergence, maximum entropy method, Kadec

AMS subject classifications. primary 46B20, 49D37; secondary 90C48

1. Introduction.

Consider the sequence of optimization problems

$$(P_n) \qquad \inf\{f(x) \mid x \in C_n\},$$

together with an associated limiting problem

$$(P_\infty) \qquad \inf\{f(x) \mid x \in C_\infty\},$$

and let us ask the question "under what conditions will optimal solutions x_n for (P_n) converge to an optimal solution x_∞ for (P_∞)?" Such questions have of course been studied extensively in the literature of well-posedness and variational convergence (see, for example, [17] and [24]). Indeed, as we shall see, it is straightforward to establish by standard techniques that when f is a strictly convex function on a Banach space, with weakly compact level sets, and the sets C_n and C_∞ are closed and convex, with C_n tending Mosco to C_∞, then, providing a suitable constraint qualification holds for (P_∞), x_n will tend weakly to x_∞. With the extra assumption that the conjugate function f^* is Fréchet differentiable whenever it has a subgradient, we obtain our goal: x_n converges to x_∞ in norm. In the language of well-posedness, this derivation is closely related to the well-known fact that the function f is *Tikhonov well posed;* in other words, every minimizing sequence converges in norm to the unique minimizer $\bar{x}$, if and only if f^* has Fréchet derivative $\bar{x}$ at 0 (see Proposition 1 in [1], and [10]).

The natural constraint qualification required is that C_∞ intersect the interior of the domain of f. We use it at two points in the argument: first, to ensure the convergence of the values of (P_n) to that of (P), and second, to deduce norm convergence of x_n from weak convergence.

Our aim in this paper is in no sense to extend this general theory, but rather to cope with a *practical* difficulty, namely, that in concrete problems the constraint qualification may easily fail. Specifically, we have in mind cases where the limit set C_∞ reduces to a single point at which the objective function f has no subgradient. For example, in best entropy estimation (see, for example, [5]), the set C_n may consist of those functions whose first n Fourier coefficients agree with those of an unknown function x_∞, and the limit set C_∞ is then simply $\{x_\infty\}$. In general, the objective function f, a measure of entropy, is a convex

* Received by the editors January 6, 1992; accepted for publication (in revised form) December 23, 1992. This work was supported in part by the Natural Sciences and Engineering Research Council of Canada.

† Department of Combinatorics and Optimization, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1.

integral functional on L_1, with domain some subset of the positive cone; typically f will have no subgradients at x_∞ unless, at the very least, x_∞ is almost everywhere strictly positive. Thus strong conditions on x_∞ are required if we wish to apply the above theory to deduce convergence of the best entropy estimates; if this unknown density can take the value 0 on regions of its domain, then we need a different approach to guarantee L_1-norm convergence.

Two refinements are necessary if we wish to remove the requirement of the constraint qualification. The first step is to ensure the weak convergence of solutions by strengthening the type of convergence of C_n to C_∞. In the next section we show how this may be accomplished by replacing the strong lim inf in the definition of Mosco convergence by the discrete lim inf. We illustrate by considering the types of moment problems that appear in maximum entropy estimation.

The section that follows contains the second step: moving from weak to norm convergence. What is needed here is the *strong rotundity* of f, in other words, the fact that f is strictly convex on its domain and has weakly compact level sets, and the fact that the *Kadec* property holds: if x_n tends weakly to x_∞ in the domain of f, and $f(x_n)$ tends to $f(x_\infty)$, then x_n tends to x_∞ in norm. The central result shows that strong rotundity of the integral functional on L_1, $f(x) := \int \phi(x(s))$, is characterized under very general conditions by the simple condition that the conjugate integrand ϕ^* is everywhere differentiable on $\mathbb{R}$ (see also [25]). Examples are given, including the Boltzmann–Shannon entropy and various other natural choices of entropy (see, for example, [7]).

2. Sequences of optimization problems. We shall suppose throughout this section that X is a Banach space, the function $f : X \to (-\infty, +\infty]$ is a proper convex function with weakly compact lower level sets, and $C_1, C_2, \ldots$ and C_∞ are closed convex subsets of X. We shall use the following limiting notions:

$$C_\infty \supset w - \overline{\lim}\, C_n := \{x \mid \exists x_{n_r} \in C_{n_r} \text{ with } x_{n_r} \to x \text{ weakly}\}. \tag{2.1}$$

$$C_\infty \subset s - \underline{\lim}\, C_n := \{x \mid \exists x_n \in C_n \text{ with } \|x_n - x\| \to 0\}. \tag{2.2}$$

$$C_\infty \subset d - \underline{\lim}\, C_n := \bigcup_{m=1}^{\infty} \bigcap_{n=m}^{\infty} C_n. \tag{2.3}$$

Conditions (2.1) and (2.2) together say that C_n converges Mosco to C_∞ [18]. Condition (2.3) is generally more restrictive than (2.2); in general,

$$\mathrm{cl}(d - \underline{\lim}\, C_n) \subset s - \underline{\lim}\, C_n$$

with equality if $d - \underline{\lim}\, C_n$ has nonempty interior [16].

We will denote the value of the problem (P_n) by $V(P_n) \in [-\infty, +\infty]$, for $n = 1, 2, \ldots, \infty$. The following easy result is standard.

LEMMA 2.4. *If* (2.1) *holds, then* $\underline{\lim}\, V(P_n) \geq V(P_\infty)$ (*finite or infinite*).

Proof. Suppose not. Then for some $M < V(P_\infty)$, there is a subsequence $x_{n_r} \in C_{n_r}$, with $f(x_{n_r}) \leq M$ for all r. Since f has compact level sets, there is a weakly convergent subsequence x_{n_r}, with limit x in $w - \overline{\lim}\, C_n$, and therefore in C_∞ by (2.1), such that $f(x) \leq M$. But then $V(P_\infty) \leq M$, which is a contradiction. □

LEMMA 2.5. *If* (2.3) *holds, then* $\overline{\lim}\, V(P_n) \leq V(P_\infty)$.

Proof. For any $x \in C_\infty$, (2.3) implies that $x \in C_n$ for all n sufficiently large, so $f(x) \geq V(P_n)$, and the result follows. □

By contrast, if we wish to weaken (2.3) to (2.2), we need to add a regularity condition, again using standard techniques.

Constraint Qualification. $\|\cdot\| - \text{int}(\text{dom } f) \cap C_\infty \neq \emptyset$. (The domain of f, dom f, is the set on which f is finite.) Clearly, if the Constraint Qualification and (2.2) hold, then for large n, a constraint qualification also holds for (P_n):

$$\|\cdot\| - \text{int}(\text{dom } f) \cap C_n \neq \emptyset.$$

LEMMA 2.6. *If* (2.2) *and the Constraint Qualification hold, then* $\overline{\lim} V(P_n) \leq V(P_\infty) < +\infty$.

Proof. By Proposition 3.3 in [19], f is continuous on the interior of its domain. Suppose x is arbitrary in int(dom f) $\cap C$. By (2.2) there exist $x_n \in C_n$ with $\|x_n - x\| \to 0$, and hence $V(P_n) \leq f(x_n) \to f(x)$. Thus $\overline{\lim} V(P_n) \leq f(x)$.

Now by the Constraint Qualification there is an x_0 in int(dom f) $\cap C_\infty$. Suppose y is arbitrary in (dom f) $\cap C_\infty$. For any λ in (0, 1],

$$\lambda x_0 + (1-\lambda)y \in C_\infty \cap \text{ int}(\text{dom } f),$$

so by the previous paragraph, $\overline{\lim} V(P_n) \leq f(\lambda x_0 + (1-\lambda)y)$. But by Theorem 10.2 in [20], $f(\lambda x_0 + (1-\lambda)y) \to f(y)$ as $\lambda \downarrow 0$, so $\overline{\lim} V(P_n) \leq f(y)$. Since y was arbitrary, the result now follows. □

Very simple examples demonstrate that the above result may fail without the Constraint Qualification. The next lemma shows how we obtain weak convergence of the optimal solutions.

LEMMA 2.7. *Suppose* (2.1) *holds,* $V(P_n) \to V(P_\infty)$, *and* (P_∞) *has a unique optimum,* x_∞. *Then for any sequence of optimal solutions* x_n *for* (P_n), $x_n \to x_\infty$ *weakly.*

Proof. Suppose the result fails, so for some ϕ in X^* and some subsequence, $\langle x_{n_r} - x_\infty, \phi\rangle \geq 1$ for all r. Now since $f(x_{n_r}) \to f(x_\infty) < +\infty$ and f has compact level sets, there is a convergent subsequence $x_{n_{r_i}} \to \bar{x}$ weakly with $f(\bar{x}) \leq f(x_\infty)$ and $\bar{x} \in C_\infty$ by (2.1). Hence, by uniqueness, $\bar{x} = x_\infty$, which contradicts $\langle x_{n_{r_i}} - x_\infty, \phi\rangle \geq 1$ for all i. □

The final lemma will allow us to move from weak to strong convergence. As usual, $f^* : X^* \to (-\infty, +\infty]$ denotes the conjugate function

$$f^*(\phi) := \sup_{x \in X} \{\langle x, \phi\rangle - f(x)\}.$$

LEMMA 2.8. *Suppose* f^* *is Fréchet differentiable on the domain of* ∂f^*. *If* $x_n \to x_\infty$ *weakly,* $f(x_n) \to f(x_\infty) < +\infty$, *and* $\partial f(x_\infty) \neq \emptyset$, *then it follows that* $\|x_n - x_\infty\| \to 0$.

Proof. Pick $\phi \in \partial f(x_\infty)$, so $x_\infty \in \partial f^*(\phi) = \{\nabla f^*(\phi)\}$ the Fréchet derivative of f^* at ϕ. Now

$$\langle x_n, \phi\rangle - f(x_n) \to \langle x_\infty, \phi\rangle - f(x_\infty) = f^*(\phi),$$

so by Proposition 1 in [1], $\|x_n - x_\infty\| \to 0$. □

The result above shows that f^* being Fréchet differentiable at any point where a subgradient exists is almost enough to guarantee the Kadec property of f; the missing condition is $\partial f(x_\infty) \neq \emptyset$. It is easily checked that the differentiability condition also implies that f is strictly convex on any convex subset of dom ∂f (see [2]; cf. Chap. 26 in [20]). Note that the converse of the above result is not true. For example, if $f(x) := \|x\|_2$ on $L_2[0, 1]$, the conclusion of the lemma is true by the Kadec property but f^* is the indicator function of the unit ball, which clearly is not Fréchet differentiable at boundary points.

We can now compare two contrasting convergence results. The first is standard, using Mosco convergence and the Constraint Qualification, whereas the second uses the stronger convergence involving the discrete lim inf and strong rotundity.

THEOREM 2.9 (i) (Mosco version). *Suppose* (2.1) *and* (2.2) *hold (in other words,* $C_n \to C_\infty$ *Mosco) and the Constraint Qualification holds. Then* $V(P_n) \to V(P_\infty) < +\infty$. *Furthermore, if* f^* *is Fréchet differentiable on the domain of* ∂f^*, *then* (P_n) *and* (P_∞) *have unique optimal solutions* x_n *and* x_∞, *respectively (for all* n *sufficiently large), and* $\|x_n - x_\infty\| \to 0$.

(ii) (Discrete version). *Suppose* (2.1) *and* (2.3) *hold. Then* $V(P_n) \to V(P_\infty)$ *(finite or infinite). Furthermore, if* $V(P_\infty) < +\infty$ *and* f *is strongly rotund, then* (P_n) *and* (P_∞) *have unique optimal solutions* x_n *and* x_∞, *respectively (for all* n *sufficiently large), and* $\|x_n - x_\infty\| \to 0$.

Proof. (i) $V(P_n) \to V(P_\infty) < +\infty$ follows from Lemmas 2.4 and 2.6. If we write δ_{C_∞} for the indicator function of C_∞, when x_∞ is optimal for (P_∞),

$$0 \in \partial(f + \delta_{C_\infty})(x_\infty) = \partial f(x_\infty) + \partial \delta_{C_\infty}(x_\infty),$$

using the Constraint Qualification and Theorem 20 in [22]. Thus, we have $\partial f(x_\infty) \neq \emptyset$, so the set of optimal solutions of (P_∞) is a convex subset of dom ∂f. Since f must be strictly convex on this set, (P_∞) has a unique optimal solution, x_∞. A similar argument using the constraint qualification for (P_n) shows that for large n (P_n) has a unique optimal solution x_n, and $x_n \to x_\infty$ weakly by Lemma 2.7. Finally, we apply Lemma 2.8 to deduce $\|x_n - x_\infty\| \to 0$.

(ii) $V(P_n) \to V(P)$ follows from Lemmas 2.4 and 2.5. Strict convexity again ensures the uniqueness of x_n and x_∞, so Lemma 2.7 once more shows weak convergence, and then strong rotundity implies convergence in norm. □

Note. In fact, due to Lemma 2.4, we could relax the strong rotundity assumption in (ii) by replacing the Kadec property by the assumption that x_n tends weakly to x_∞ and $f(x_n) \uparrow f(x_\infty)$ implies that x_n tends to x_∞ in norm.

Example. If $C_1 \supset C_2 \supset C_3 \supset \cdots$ and $C_\infty = \bigcap_1^\infty C_n$, then (2.1) and (2.3) hold. Indeed, it is almost immediate that

$$w - \overline{\lim}\, C_n = \bigcap_{n=1}^{\infty} C_n = d - \underline{\lim}\, C_n.$$

Our second example is modeled on sequences of moment problems. In best entropy estimation we seek to estimate an unknown density given some of its moments by choosing that density (perhaps within some tolerance) satisfying the known moment conditions which minimizes a certain objective function (a measure of entropy). We seek conditions ensuring that the estimates converge in norm to the unknown density as the amount of moment information increases.

Example. Suppose the functions $a_1, a_2 \ldots$ are weak-star densely spanning in $L_\infty(S, \mu)$ and $\bar{x} \in L_1$. Suppose that for each i, n, F_i^n is a closed subset of $\mathbb{R}$, containing $\langle a_i, \bar{x} \rangle$, whose diameter tends to 0 as $n \to \infty$. Suppose finally that

$$C_n := \{x \in L_1 \mid \langle a_i, x \rangle \in F_i^n \text{ for all } i\}.$$

(Thus, C_n consists of those densities whose moments agree with those of $\bar{x}$ within certain tolerances.) Then $w - \overline{\lim}\, C_n = \{\bar{x}\} = d - \underline{\lim}\, C_n$.

To see this, note that certainly, since $\bar{x} \in C_n$ for all $n, \bar{x} \in w - \overline{\lim} C_n$ and $\bar{x} \in d - \underline{\lim} C_n$. On the other hand, if $x_{n_r} \in C_{n_r}$ and $x_{n_r} \to x$ weakly, for any $i, \langle a_i, x_{n_r} - \bar{x} \rangle \leq \operatorname{diam} F_i^{n_r} \to 0$ as $r \to \infty$, so $\langle a_i, x - \bar{x} \rangle = 0$. Thus, $x = \bar{x}$, which demonstrates the first equality.

Finally, if $x \in d - \underline{\lim} C_n$, then $x \in C_n$ for all n sufficiently large, so $\langle a_i, x - \bar{x} \rangle \leq \operatorname{diam} F_i^n \to 0$ as $n \to \infty$, which shows that $x = \bar{x}$. Hence, the second equality.

3. Strong rotundity. We shall suppose throughout that (S, μ) is a complete finite measure space (with nonzero μ), and $\phi : \mathbb{R} \to (-\infty, +\infty]$ is a proper, closed, convex function. We denote the interior of the domain of ϕ (assumed nonempty) by (α, β), where $-\infty \leq \alpha < \beta \leq +\infty$. Since ϕ is a normal convex integrand, we can define the proper weakly lower semicontinuous functional $I_\phi : L_1(S, \mu) \to (-\infty, +\infty]$ by $I_\phi(x) := \int_S \phi(x(s)) d\mu(s)$, with conjugate $I_\phi^* : L_\infty(S, \mu) \to (-\infty, +\infty]$ given by $I_\phi^* = I_{\phi^*}$, where ϕ^* is the conjugate of ϕ (see [22]).

LEMMA 3.1. *I_ϕ is strictly convex on its domain if and only if ϕ is strictly convex on its domain.*

Proof. It is well known and straightforward that I_ϕ is strictly convex if ϕ is. The converse follows by considering constant functions. □

LEMMA 3.2. *I_{ϕ^*} is Fréchet differentiable everywhere on $L_\infty(S, \mu)$ if and only if ϕ^* is differentiable everywhere on $\mathbb{R}$.*

Proof. Suppose ϕ^* is differentiable on $\mathbb{R}$, so by Theorem 25.5 in [20] it is continuously differentiable. Given any $y \in L_\infty(S, \mu)$, pick m and M in $\mathbb{R}$ with $m \leq y(s) \leq M$ almost everywhere. Since $(\phi^*)'$ is uniformly continuous on $[m-1, M+1]$, for almost every s, given any $\epsilon > 0$, there is a $\delta > 0$ such that $|(\phi^*)'(y(s) + v) - (\phi^*)'(y(s))| < \epsilon$ whenever $|v| < \delta$. Thus, by the Mean Value Theorem, for some $v' \in (-\delta, \delta)$

$$\begin{aligned} &|\phi^*(y(s) + v) - \phi^*(y(s)) - v(\phi^*)'(y(s))| \\ &\quad = |v| \, |(\phi^*)'(y(s) + v') - (\phi^*)'(y(s))| \\ &\quad \leq \epsilon |v|, \end{aligned}$$

so if $\|h\|_\infty \leq \delta$,

$$\left| I_{\phi^*}(y + h) - I_{\phi^*}(y) - \int_S h(s)(\phi^*)'(y(s)) d\mu(s) \right| \leq \epsilon \|h\|_\infty.$$

This demonstrates that I_{ϕ^*} has Fréchet derivative $\nabla I_{\phi^*}(y) = (\phi^*)'(y(\cdot))$ at y. The converse follows by considering constant functions. □

LEMMA 3.3. *If $x \in L_\infty(S, \mu)$ with* ess inf $x > \alpha$ *and* ess sup $x < \beta$, *then $\partial I_\phi(x) \neq \emptyset$.*

Proof. Denote the essential infimum and supremum of x by m and M, respectively. The multifunction $\partial\phi : [m, M] \to \mathbb{R}$ is upper semicontinuous [15]. In other words, for any closed set F in $\mathbb{R}$,

$$(\partial\phi)_F := \{u \in [m, M] \mid \partial\phi(u) \cap F \neq \emptyset\}$$

is closed. Thus, the multifunction $s \mapsto \partial\phi(x(s))$ is measurable, since

$$\{s \in S \mid \partial\phi(x(s)) \cap F \neq \emptyset\} = \{s \in S \mid x(s) \in (\partial\phi)_F\}$$

is measurable for any closed F. Since this multifunction has nonempty closed (actually compact) values, there is a measurable selection $y(s) \in \partial\phi(x(s))$ almost everywhere, by Theorem 14.2.2 in [15]. Since $-\infty < \min \partial\phi(m) \leq y(s) \leq \max \partial\phi(M) < +\infty$ almost everywhere, it follows that $y \in L_\infty(S, \mu)$; so, directly from the definition, $y \in \partial I_\phi(x)$. □

For any m, define (for a fixed $x \in L_1(S,\mu)$)

$$S_m := \left\{ s \in S \mid (-m) \vee \left(\alpha + \frac{1}{m}\right) \le x(s) \le \left(\beta - \frac{1}{m}\right) \wedge m \right\}.$$

LEMMA 3.4. *If $\alpha < x(s) < \beta$ almost everywhere, then $\mu(S_m^c) \downarrow 0$ as $m \to \infty$.*

Proof. The sets S_m are nested and increasing with

$$\bigcup_{m=1}^{\infty} S_m = \{s \mid \alpha < x(s) < \beta\}.$$

Thus, $\mu(S_m) \uparrow \mu(S)$ as $m \to \infty$. □

For any measurable subset T of S, we denote the restrictions of μ and x to T by $\mu|_T$ and $x|_T$, and define $I_\phi^T : L_1(T, d\mu|_T) \to (-\infty, +\infty]$ by $I_\phi^T(z) := \int_T \phi(z(s))d\mu(s)$.

LEMMA 3.5. *Suppose $x_n \to x$ weakly in $L_1(S,\mu)$ and $I_\phi(x_n) \to I_\phi(x) < +\infty$. Then for any measurable subset T of S, $x_n|_T \to x|_T$ weakly in $L_1(T, \mu|_T)$ and $I_\phi^T(x_n|_T) \to I_\phi^T(x|_T) < +\infty$.*

Proof. The weak convergence of $x_n|_T$ to $x|_T$ follows by integrating with respect to functions zero on T^c. Since I_ϕ^T and $I_\phi^{T^c}$ are both weakly lower semicontinuous, we have $\underline{\lim} I_\phi^T(x_n|_T) \ge I_\phi^T(x|_T)$ and $\underline{\lim} I_\phi^{T^c}(x_n|_{T^c}) \ge I_\phi^{T^c}(x|_{T^c})$ (since we also have $x_n|_{T^c} \to x|_{T^c}$ weakly in $L_1(T^c, \mu|_{T^c})$). On the other hand, since $I_\phi(z) = I_\phi^T(z|_T) + I_\phi^{T^c}(z|_{T^c})$ for any z,

$$\begin{aligned} \limsup I_\phi^T(x_n|_T) &= \limsup(I_\phi(x_n) - I_\phi^{T^c}(x_n|_{T^c})) \\ &= I_\phi(x) - \liminf I_\phi^{T^c}(x_n|_{T^c}) \\ &\le I_\phi(x) - I_\phi^{T^c}(x|_{T^c}) \\ &= I_\phi^T(x|_T), \end{aligned}$$

and the result follows. □

LEMMA 3.6. *Suppose ϕ^* is differentiable everywhere on $\mathbb{R}$. If $x_n \to x$ weakly in $L_1(s,\mu)$, $I_\phi(x_n) \to I_\phi(x) < +\infty$, and $\alpha < x(s) < \beta$ almost everywhere, then it follows that $\|x_n - x\|_1 \to 0$.*

Proof. Since $x_n \to x$ weakly, $(x_n)_1^\infty \cup \{x\}$ is weakly compact in $L_1(S,\mu)$. Given any $\epsilon > 0$, by the Dunford–Pettis criterion [9] there exists a $\delta > 0$ such that if $\mu(T) \le \delta$, $\int_T |x_n(s)|d\mu(s) < \epsilon$ for all n and $\int_T |x(s)|d\mu(s) < \epsilon$. By Lemma 3.4 there is an m with $\mu(S_m^c) \le \delta$, so $\int_{S_m^c} |x_n(s) - x(s)|d\mu(s) < 2\epsilon$ for all n.

Now by Lemma 3.5, as $n \to \infty, x_n|_{S_m} \to x|_{S_m}$ weakly in $L_1(S_m, \mu|_{S_m})$ and $I_\phi^{S_m}(x_n|_{S_m}) \to I_\phi^{S_m}(x|_{S_m}) < +\infty$, and certainly $x|_{S_m} \in L_\infty(S_m, \mu|_{S_m})$ with

$$\text{ess inf } x|_{S_m} > \alpha \quad \text{and} \quad \text{ess sup } x|_{S_m} < \beta.$$

Thus, by Lemma 3.3, $\partial I_\phi^{S_m}(x|_{S_m}) \ne \emptyset$, so we can apply Lemma 2.8 (on $L_1(S|_m, \mu|_{S_m})$) to deduce that $\int_{S_m} |x_n(s) - x(s)|d\mu(s) \to 0$ as $n \to \infty$.

Finally, since for all n,

$$\begin{aligned} \|x_n - x\|_1 &= \int_{S_m} |x_n(s) - x(s)|d\mu(s) + \int_{S_m^c} |x_n(s) - x(s)|d\mu(s) \\ &< \int_{S_m} |x_n(s) - x(s)|d\mu(s) + 2\epsilon, \end{aligned}$$

we obtain $\limsup \|x_n - x\|_1 \leq 2\epsilon$. As ϵ was arbitrary, the result follows. □

THEOREM 3.7. *Suppose ϕ^* is differentiable everywhere on $\mathbb{R}$. If $x_n \to x$ weakly in $L_1(S,\mu)$ and $I_\phi(x_n) \to I_\phi(x) < +\infty$, then $\|x_n - x\|_1 \to 0$.*

Proof. Since $I_\phi(x) < +\infty, \alpha \leq x(s) \leq \beta$ almost everywhere and for all n sufficiently large $\alpha \leq x_n(s) \leq \beta$ almost everywhere. Define

$$\begin{aligned} S^\gamma &:= \{s \in S \mid \alpha < x(s) < \beta\}, \\ S^\alpha &:= \{s \in S \mid x(s) = \alpha\}, \\ S^\beta &:= \{s \in S \mid x(s) = \beta\}. \end{aligned}$$

By Lemma 3.5, $x_n|_{S^\gamma} \to x|_{S^\gamma}$ weakly in $L_1(S^\gamma, \mu|_{S^\gamma})$, and

$$I_\phi^{S^\gamma}(x_n|_{S^\gamma}) \to I_\phi^{S^\gamma}(x|_{S^\gamma}) < +\infty,$$

so applying Lemma 3.6, $\int_{S^\gamma} |x_n(s) - x(s)| d\mu(s) \to 0$. But now, for all n sufficiently large,

$$\begin{aligned} \|x_n - x\|_1 &= \int_{S^\gamma} |x_n(s) - x(s)| d\mu(s) \\ &\quad + \int_{S^\alpha} |x_n(s) - x(s)| d\mu(s) + \int_{S^\beta} |x_n(s) - x(s)| d\mu(s) \\ &= \int_{S^\gamma} |x_n(s) - x(s)| d\mu(s) + \int_{S^\alpha} (x_n(s) - \alpha) d\mu(s) \\ &\quad + \int_{S^\beta} (\beta - x_n(s)) d\mu(s) \\ &\quad \to 0, \end{aligned}$$

as $n \to \infty$. □

We can now prove the central result.

THEOREM 3.8. *If ϕ^* is differentiable everywhere on $\mathbb{R}$, then $I_\phi(\cdot)$ is strongly rotund on $L_1(S,\mu)$. The converse is also true if (S,μ) is not purely atomic.*

Proof. If ϕ^* is differentiable everywhere on $\mathbb{R}$, it is essentially smooth, so ϕ is essentially strictly convex by Theorem 26.3 in [20]. On $\mathbb{R}$ this is equivalent to ϕ being strictly convex on its domain, and hence I_ϕ is strictly convex on its domain (Lemma 3.1). Since ϕ is everywhere finite, I_ϕ has weakly compact level sets by Corollary 2B in [21]. Finally, Theorem 3.7 shows the Kadec property.

Conversely, using Lemma 3.1, strict convexity of ϕ and hence essential smoothness of ϕ^* follows from the strict convexity of I_ϕ, whereas I_ϕ having compact level sets implies that ϕ^* is everywhere finite by Theorem 2.10 in [5]. □

Note. In fact the above argument shows that if ϕ^* is differentiable everywhere on $\mathbb{R}$, then for any y in $L_\infty(S,\mu)$, the functional $I_\phi(\cdot) + \langle y, \cdot \rangle$ is strongly rotund.

The approach we have taken has been to work as far as possible via Fréchet differentiability of the conjugate. A different, more geometric technique may be found in [25], and another independent approach in [23]. Theorem 3 in [25] shows that the hypotheses of Theorem 3.7 also give that $\|\phi(x_n(\cdot)) - \phi(x(\cdot))\|_1 \to 0$.

As suggested in the previous section, our interest in strongly rotund functions arose from the study of best entropy estimation for moment problems. Many of the functions ϕ that are used in practice may be described in a unified framework based on a Bayesian statistical interpretation (see [7]). Each such choice of ϕ is associated with a unique "prior" nonnegative measure ρ on $\mathbb{R}$, via conjugation and (two-sided) Laplace transformation:

$$\phi^*(v) = \log \int_{-\infty}^{\infty} e^{uv} d\rho(u). \tag{3.9}$$

It may be easily checked from Hölder's inequality that ϕ^* is strictly convex (providing ρ is not supported on a single point) and is differentiable on the interior of its domain. In particular, when ϕ^* is everywhere finite our results show that I_ϕ is strongly rotund, implying convergence in L_1 norm of the associated estimates. Some examples follow:

(i) *Normal distribution* $d\rho = \frac{1}{\sqrt{2\pi}} e^{-u^2/2} du$. We obtain $\phi^*(v) = \frac{1}{2}v^2$, so $I_\phi(x) = \frac{1}{2}\|x\|_2^2$.

(ii) *Poisson distribution* $\rho = \frac{1}{e}\sum_{r=0}^{\infty} \frac{1}{r!}\delta_{\{r\}}$. We obtain $\phi^*(v) = e^{v-1}$, so $I_\phi(x) = \int x \log x$, which is (minus) the Boltzmann–Shannon entropy. The strong rotundity in this case was shown directly in [4].

(iii) *Binomial distribution* $\rho = \frac{1}{2}(\delta_0 + \delta_1)$. We obtain $\phi^*(v) = \log\left(\frac{1}{2}(1 + e^v)\right)$, and

$$\phi(u) = \begin{cases} u \log u + (1-u)\log(1-u) + \log 2 & \text{if } u \in (0,1), \\ +\infty & \text{otherwise,} \end{cases}$$

which is, up to a constant, (minus) the Fermi–Dirac entropy. It is curious that the general binomial distribution leads to the same entropy.

(iv) *Lebesgue measure.*

$$d\rho = \begin{cases} du & \text{on } \mathbb{R}_+, \\ 0 & \text{on } \mathbb{R}_-. \end{cases}$$

We obtain

$$\phi^*(v) = \begin{cases} -\log(-v) & \text{if } v < 0, \\ +\infty & \text{if } v \geq 0, \end{cases}$$

so

$$\phi(u) = \begin{cases} -\log(u) - 1 & \text{if } u > 0, \\ +\infty & \text{if } u \leq 0, \end{cases}$$

and I_ϕ is the Burg entropy [6]. In this case, since ϕ^* is not everywhere finite, I_ϕ does not have weakly compact level sets.

Not all of the entropies appearing in the literature may be encompassed in this framework. For example, the choice

$$\phi(u) := \begin{cases} \frac{1}{p}u^p & \text{if } u \geq 0, \\ +\infty & \text{if } u < 0 \end{cases} \tag{3.10}$$

is made in "L_p spectral estimation" $(1 < p < +\infty)$ (see [3] and [11]) and then $\phi^*(v) := \frac{1}{q}v_+^q$, where $\frac{1}{p} + \frac{1}{q} = 1$. Since ϕ^* is not strictly convex, it cannot be written in the form (3.9). Our results show that I_ϕ is strongly rotund in L_1, and in fact more is clearly true: since the unit ball in L_p is weakly compact and $\|\cdot\|_p$ is Kadec [9], I_ϕ is actually strongly rotund in L_p. Notice that when $p = 1$ in (3.10), we obtain

$$\phi^*(v) = \begin{cases} 0 & \text{if } v \leq 1, \\ +\infty & \text{if } v > 1, \end{cases}$$

which is not everywhere finite, and thus I_ϕ does not have weakly compact level sets (not surprisingly, since the unit ball in L_1 is not weakly compact).

4. Existence of strongly rotund functions. On any measure space (S, μ), if $1 < p < +\infty$, then $\|\cdot\|_p^p$ is a strongly rotund functional on $L_p(S)$ (see, for example, [9]), which is everywhere finite. Thus, as we have seen, sequences of optimization problems whose objective function is an L_p norm are particularly well behaved. Of course, the L_1 norm will not have this property; the results below show that the existence of everywhere finite, strongly rotund functions is characterized by reflexivity. The Boltzmann–Shannon entropy, $\int x \log x$, may be thought of as a surrogate reflexive norm on the positive cone in L_1 due to its strong rotundity.

LEMMA 4.1. *If V is a vector space, $f : V \to (-\infty, +\infty]$ is a convex function, and $\alpha \in (\inf f, +\infty)$, then* $\operatorname{dom} f \subset \operatorname{aff}\{v \in V \mid f(v) \leq \alpha\}$.

Proof. Pick any v_0 with $f(v_0) < \alpha$, and suppose $v_1 \in \operatorname{dom} f$. Providing $\lambda > 0$ is sufficiently small,

$$\begin{aligned} f((1-\lambda)v_0 + \lambda v_1) &\leq (1-\lambda)f(v_0) + \lambda f(v_1) \\ &= f(v_0) + \lambda(f(v_1) - f(v_0)) \\ &\leq \alpha, \end{aligned}$$

and since $v_1 = (1/\lambda)((1-\lambda)v_0 + \lambda v_1) + (1 - (1/\lambda))v_0$, the result follows. □

THEOREM 4.2. *Suppose that X is a Banach space and $f : X \to (-\infty, +\infty]$ is a convex function with weakly compact level sets, whose domain is spanning. Then X is reflexive.*

Proof. Choose any $\alpha \in (\inf f, +\infty)$ and define $L := \{x \in X \mid f(x) \leq \alpha\}$, which, by assumption, is weakly compact. Thus, the closed, absolutely convex hull of L, $\overline{\operatorname{aco}}\, L$, is also weakly compact [13, p. 162]. But now we have

$$X = \operatorname{span}(\operatorname{dom} f) \subset \operatorname{span}(\operatorname{aff} L) = \operatorname{span} L \subset \operatorname{span}(\overline{\operatorname{aco}}\, L) = \bigcup_{n=1}^{\infty} n(\overline{\operatorname{aco}}\, L)$$

using Lemma 4.1. Now by Baire category, $\overline{\operatorname{aco}}\, L$ has nonempty interior, so for some $r > 0$ and $\bar{x}$ in X the closed ball $\bar{B}(\bar{x}; r) \subset \overline{\operatorname{aco}}(L)$, and so is weakly compact. Thus X is reflexive [13, p. 126]. □

Suppose that we wished to select an objective function for solving sequences of moment problems that would guarantee convergence in the norm $\|\cdot\|_1$ of estimates to an arbitrary unknown, nonnegative, L_1 density. To ensure convergence of the estimates using our methods, we would require a strongly rotund functional on the space $L_1(S, \mu)$ whose domain contained the positive cone and hence was spanning. Since $L_1(S, \mu)$ is not reflexive unless the measure space (S, μ) decomposes as a finite set of atoms, the above result shows this to be impossible: since the level sets of the objective function cannot be weakly compact, we cannot even ensure attainment.

Example. Suppose X is a reflexive Banach space. Then X has a renorming $\|\cdot\|$ that is Fréchet differentiable and locally uniformly convex (and whose dual norm has the same properties) [8, p. 167]. It is then easy to check that $f(x) := \|x\|^2$ is everywhere finite and strongly rotund.

COROLLARY 4.3. *Suppose X is a Banach space. Then there exists an everywhere finite strongly rotund function $f : X \to \mathbb{R}$ if and only if X is reflexive.*

Suppose instead that we ask just for a strongly rotund functional on $L_1(S, \mu)$ whose domain is *dense*. Assuming (S, μ) is finite, we could use $f(x) := \int x(s)^2 ds$. This is a consequence of the following general construction.

THEOREM 4.4. *Suppose X and Y are normed spaces, $T : X \to Y$ is continuous and linear, and $f : X \to (-\infty, +\infty]$ is strongly rotund. Define $g : Y \to (-\infty, +\infty]$ by $g(y) := \inf\{f(x) \mid Tx = y\}$. Then g is strongly rotund.*

Proof. Because the level sets of f are weakly compact, for any x in $\mathbb{R}$, $g(y) \leq \alpha$ if and only if there exists an x with $Tx = y$ and $f(x) \leq \alpha$, so

$$\{y \in Y \mid g(y) \leq \alpha\} = T\{x \in X \mid f(x) \leq \alpha\}.$$

Thus the level sets of g are weakly compact. A similar calculation shows that dom $g = T(\text{dom } f)$.

Now suppose that $y_1 \neq y_2$ in dom g and $\lambda \in (0,1)$. By compactness, $g(y_i) = f(x_i)$ for some x_i with $Tx_i = y_i, i = 1, 2$, and because

$$T(\lambda x_1 + (1-\lambda)x_2) = \lambda y_1 + (1-\lambda)y_2,$$

we have

$$\begin{aligned} g(\lambda y_1 + (1-\lambda)y_2) &\leq f(\lambda x_1 + (1-\lambda)x_2) \\ &< \lambda f(x_1) + (1-\lambda)f(x_2) \\ &= \lambda g(y_1) + (1-\lambda)g(y_2). \end{aligned}$$

Thus g is strictly convex.

It remains to demonstrate the Kadec property, so suppose $y_n \to y_\infty$ weakly in Y and $g(y_n) \to g(y_\infty) < +\infty$. By compactness, $g(y_n) = f(x_n)$ for some x_n with $Tx_n = y_n, n = 1, 2, \ldots, \infty$. Pick an arbitrary subsequence (y_{n_i}). Since f has weakly compact level sets, there is a subsequence $(x_{n_{i_j}})$ converging weakly to some $\bar{x}$ in X. Now note that

$$T\bar{x} = \text{w-lim } Tx_{n_{i_j}} = \text{w-lim } y_{n_{i_j}} = y_\infty,$$

and so, by the lower semicontinuity of f,

$$g(y_\infty) \leq f(\bar{x}) \leq \underline{\lim} f(x_{n_{i_j}}) = \underline{\lim} g(y_{n_{i_j}}) = g(y_\infty).$$

Thus $g(y_\infty) = f(\bar{x})$, and since f is strictly convex, $\bar{x} = x_\infty$. Now we have $x_{n_{i_j}} \to x_\infty$ weakly in X and $f(x_{n_{i_j}}) \to f(x_\infty) < +\infty$, so by the strong rotundity of f, $\|x_{n_{i_j}} - x_\infty\| \to 0$, and thus $\|y_{n_{i_j}} - y_\infty\| \to 0$ by continuity. Since (y_{n_i}) was an arbitrary subsequence, we deduce that $\|y_n - y_\infty\| \to 0$ as required. □

COROLLARY 4.5. *Suppose (S, μ) is a finite measure space, $1 < r < +\infty, 1 \leq p \leq r$, and $f : L_r(S, \mu) \to (-\infty, +\infty]$ is strongly rotund. Then the functional $g : L_p(S, \mu) \to (-\infty, +\infty]$ defined by*

$$g(y) := \begin{cases} f(y) & \text{if } y \in L_r, \\ +\infty & \text{otherwise}, \end{cases}$$

is strongly rotund.

Proof. Let $T : L_r \to L_p$ be the (continuous) embedding, and apply Theorem 4.4. □

Example. For any $1 < r < +\infty$, $\|\cdot\|_r^r$ is strongly rotund on $L_r(S, \mu)$, so providing (S, μ) is finite, $g(y) := \int y(s)^r ds$ will be strongly convex on $L_p(S, \mu)$ for any $1 \leq p \leq r$.

Example. Let $X := L_1[0,1] \times \mathbb{R}, Y := C[0,1]$, and then proceed to define $T : X \to Y$ by $(T(x;r))(t) := r + \int_0^t x(s)ds$. If we define $f : X \to (-\infty, +\infty]$ by

$$f(x;r) := \int_0^1 \phi(x(s))ds + \phi(r),$$

where ϕ is (minus) the Boltzmann–Shannon entropy,

$$\phi(u) := \begin{cases} u \log u & \text{if } u > 0, \\ 0 & \text{if } u = 0, \\ +\infty & \text{if } u < 0, \end{cases}$$

then f is strongly rotund by Theorem 3.8.

It is now easy to verify that the functional $g : C[0,1] \to (-\infty, +\infty]$ of Theorem 4.4 is given by

$$g(y) := \begin{cases} \displaystyle\int_0^1 \phi\left(\frac{dy}{ds}\right) ds + \phi(y(0)), & \text{if } \frac{dy}{ds} \in L_1[0,1], \\ +\infty, & \text{otherwise,} \end{cases}$$

and so g is strongly rotund.

Now consider the sequence of moment problems

$$(M_n) \begin{cases} \inf & g(y) \\ \text{subject to} & \langle a_i, y \rangle = \langle a_i, \bar{y} \rangle, \quad i = 1, \ldots, n, \\ & y \in C[0,1], \end{cases}$$

where the functions $a_1, a_2, \ldots,$ are weak-star densely spanning in $M[0,1]$, the regular Borel measures, and $\bar{y}$ is a fixed, monotonically increasing, nonnegative function. Then, providing $g(\bar{y}) < +\infty$ (which will be the case, for example, if $\bar{y}$ is Lipschitz on [0,1]), we can apply Theorem 2.9(ii) to deduce the uniform convergence of the optimal solutions of (M_n) to $\bar{y}$.

The construction in Theorem 4.4 also allows us to characterize those spaces on which there exist strongly rotund functions with dense domain. We say a normed space is *weakly compactly generated* (WCG) if there exists a weakly compact convex set whose linear span is dense (see [8]). Any reflexive space is WCG, as is any separable space. For any σ-finite measure space (S, μ), $L_1(S, \mu)$ is WCG [8, p. 143].

THEOREM 4.6. *Suppose Y is a Banach space and $f : Y \to (-\infty, +\infty]$ is a convex function with weakly compact level sets, whose domain is densely spanning. Then Y is WCG.*

Proof. Choose any α in $(\inf f, +\infty)$ and define $L := \{y \in Y \mid f(y) \leq \alpha\}$. By Lemma 4.1, span(dom f) $\subset$ span L, so the weakly compact, convex set L is densely spanning, as required. □

Suppose we wished to select an objective function for solving sequences of moment problems that would guarantee convergence in $\| \cdot \|_\infty$ of estimates to an unknown density. To ensure convergence using our methods to even a dense subset of the possible nonnegative, essentially bounded, unknown densities, we would require a strongly convex functional on $L_\infty(S, \mu)$ whose domain is dense in the positive cone. The above result demonstrates that this is impossible, since $L_\infty(S, \mu)$ is not WCG unless (S, μ) decomposes into a finite number of atoms. To see this, note that in any other case $L_1(S, \mu)$ contains a copy of l_1 as a subspace. Since any subspace of an Asplund space is Asplund, and l_1 is not Asplund, $L_1(S, \mu)$ is not an Asplund space, and hence its dual, $L_\infty(S, \mu)$, is not WCG [19].

By contrast, if we choose

$$\phi(u) = \begin{cases} e^u, & u \geq 0, \\ +\infty, & u < 0, \end{cases}$$

then using I_ϕ as the objective functional will ensure convergence in $\| \cdot \|_p$ to any nonnegative, essentially bounded, unknown density for every $p < +\infty$, provided (S, μ) is finite (see [5]).

THEOREM 4.7. *Suppose Y is a Banach space. Then there is a strongly rotund function on Y with dense domain if and only if Y is WCG.*

Proof. By the Davis–Figiel–Johnson–Pekzynski factorization theorem (see, for example, [9]), if Y is WCG there is a reflexive Banach space X and a continuous linear map $T : X \to Y$ with dense range. By Corollary 4.3 there is an everywhere finite strongly rotund function f on X. The function g of Theorem 4.4 is then strongly rotund on Y and has domain $T(\operatorname{dom} f)$, which is dense. The converse follows by Theorem 4.6. □

We end with examples of weak-star (weak*) strongly rotund functions: strictly convex functions $f : Y^* \to (-\infty, +\infty]$ on a dual space Y^* whose level sets are weak*-compact and such that $\theta_n \to \theta$ weak* and $f(\theta_n) \to f(\theta)$ implies $\|\theta_n - \theta\|_* \to 0$.

Let Γ be any nonempty set. Then $c_0(\Gamma)$ is WCG [8, p. 143], and so has an equivalent norm $|||\cdot|||$ whose dual norm $|||\cdot|||_*$ on $l_1(\Gamma)$ is strictly convex [8, p. 167]. The standard norm $\|\cdot\|_1$ on $l_1(\Gamma)$ has the weak* Kadec property, by Theorem 13.47 in [12]: if $x_n \to x$ weak* and $\|x_n\|_1 \to \|x\|_1$, then $\|x_n - x\|_1 \to 0$.

Now define a new norm on $l_1(\Gamma)$ by $|x| := |||x|||_* + \|x\|_1$. This norm is strictly convex, since $|||\cdot|||_*$ is, and it has weak*-compact level sets, since both $|||\cdot|||_*$ and $\|\cdot\|_1$ are dual norms and have weak*-compact balls. If we now define $f : l_1(\Gamma) \to \mathbb{R}$ by $f(x) = \frac{1}{2}|x|^2$, then it is clear that f is strictly convex with weak*-compact level sets. To see that it is actually weak* strongly rotund, suppose $x_n \to x$ weak* and $f(x_n) \to f(x)$. By weak* lower-semicontinuity, $\underline{\lim}\|x_n\|_1 \geq \|x\|_1$, while $\underline{\lim}|||x_n|||_* \geq |||x|||_*$, so

$$\overline{\lim}\|x_n\|_1 = \overline{\lim}(|x_n| - |||x_n|||_*) = |x| - \underline{\lim}|||x_n|||_* \leq |x| - |||x|||_* = \|x\|_1.$$

Thus, $\|x_n\|_1 \to \|x\|_1$, so $\|x_n - x\|_1 \to 0$ as required.

Notice that $f^* : l_\infty(\Gamma) \to \mathbb{R}$ is given by $f^*(z) = \frac{1}{2}|z|_*^2$, where $|\cdot|_*$ is the dual norm. Thus, f^* is certainly not Fréchet differentiable everywhere, since this would imply that $|\cdot|_*$ was Fréchet and so $l_1(\Gamma)$ would have to be reflexive [8, p. 34], which fails whenever Γ is infinite.

In general, whenever X is a separable Banach space with a separable dual, X can be renormed (by $\|\cdot\|$, say) in such a way that the dual norm $\|\cdot\|_*$ is strictly convex and has the weak* Kadec property [14]. It then follows that $\frac{1}{2}\|\cdot\|_*^2$ is weak* strongly rotund.

REFERENCES

[1] E. ASPLUND, *Fréchet differentiability of convex functions*, Acta Math., 121 (1968), pp. 31–47.

[2] E. ASPLUND AND R. T. ROCKAFELLAR, *Gradients of convex functions*, Trans. Amer. Math. Soc., 139 (1969), pp. 443–467.

[3] A. BEN-TAL, J. M. BORWEIN, AND M. TEBOULLE, *A dual approach to multidimensional L_p spectral estimation problems*, SIAM J. Control Optim., 26 (1988), pp. 985–996.

[4] J. M. BORWEIN AND A. S. LEWIS, *Convergence of best entropy estimates*. SIAM J. Optim., 1 (1991), pp. 191–205.

[5] ———, *On the convergence of moment problems*, Trans. Amer. Math. Soc., 325 (1991), pp. 249–271.

[6] J. P. BURG, *Maximum entropy spectral analysis*, paper presented at 37th Meeting of the Society of Exploration Geophysicists, Oklahoma City, 1967.

[7] D. DACUNHA-CASTELLE AND F. GAMBOA, *Maximum d'entropie et problème des moments*, Ann. Inst. H. Poincaré, 26 (1990), pp. 567–596.

[8] J. DIESTEL, *Geometry of Banach Spaces—Selected Topics*, Lecture Notes in Mathematics 485, Springer-Verlag, New York, 1975.

[9] ———, *Sequences and Series in Banach Spaces*, Springer-Verlag, New York, 1984.

[10] A. DONTCHEV AND T. ZOLEZZI, *Well-Posed Optimization Problems*, Springer-Verlag, New York, 1993.

[11] R. K. GOODRICH AND A. STEINHARDT, *L_2 spectral estimation*, SIAM J. Appl. Math., 46 (1986), pp. 417–428.

[12] E. HEWITT AND K. STROMBERG, *Real and Abstract Analysis*, Springer-Verlag, New York, 1975.

[13] R. B. HOLMES, *Geometric Functional Analysis and Its Applications*, Springer-Verlag, New York, 1975.

[14] V. L. KLEE, *Mappings into normed linear spaces*, Fund. Math., 49 (1960), pp. 25–34.

[15] E. KLEIN AND A. C. THOMPSON, *Theory of Correspondences*, Wiley, New York, 1984.

[16] A. S. Lewis and R. Lucchetti, Private communication, 1991.
[17] R. Lucchetti and F. Patrone, *Hadamard and Tyhonov well-posedness of a certain class of convex functions*, J. Math. Anal. Appl., 88 (1982), pp. 204–215.
[18] U. Mosco, *Convergence of convex sets and solutions of variational inequalities*, Adv. Math., 3 (1969), pp. 510–585.
[19] R. R. Phelps, *Convex Functions, Monotone Operators and Differentiability*, Lecture Notes in Mathematics 1364, Springer-Verlag, New York, 1989.
[20] R. T. Rockafellar, *Convex Analysis*, Princeton University Press, Princeton, NJ, 1970.
[21] ———, *Integrals which are convex functionals*, II, Pacific J. Math., 39 (1971), pp. 439–469.
[22] ———, *Conjugate Duality and Optimization*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1974.
[23] M. Teboulle and I. Vajda, *Convergence of best ϕ-entropy estimates*, IEEE Trans. Inform. Theory, 39 (1993), pp. 297–301.
[24] A. N. Tyhonov, *On the stability of the functional optimization problem*, U.S.S.R. Comput. Math. and Phys., 6 (1966), pp. 28–33.
[25] A. Visintin, *Strong convergence results related to strict convexity*, Comm. Partial Differential Equations, 9 (1984), pp. 439–466.

SIAM J. OPTIMIZATION
Vol. 4, No. 1, pp. 159–176, February 1994
© 1994 Society for Industrial and Applied Mathematics
009

LOCAL MINIMIZERS OF QUADRATIC FUNCTIONS ON EUCLIDEAN BALLS AND SPHERES*

JOSÉ MARIO MARTÍNEZ†

Abstract. In this paper a characterization of the local-nonglobal minimizer of a quadratic function defined on a Euclidean ball or a sphere is given. It is proven that there exists at most one local-nonglobal minimizer and that the Lagrange multiplier that corresponds to this minimizer is the largest solution of a nonlinear scalar equation. An algorithm is proposed for computing the local-nonglobal minimizer.

Key words. quadratic constraints, trust regions, local minimizers

AMS subject classifications. 65K10, 90C20

1. Introduction.

In this paper we consider the following problems:

$$\begin{aligned} &\text{Minimize } q(s) \\ &\text{s.t. } ||s|| \leq \Delta \end{aligned} \tag{1.1}$$

and

$$\begin{aligned} &\text{Minimize } q(s) \\ &\text{s.t. } ||s|| = \Delta, \end{aligned} \tag{1.2}$$

where

$$q(s) = \tfrac{1}{2} s^T G s + g^T s, \tag{1.3}$$

$\Delta > 0, s, g \in \mathbb{R}^n, G = G^T$, and $||.||$ is the Euclidean norm on $\mathbb{R}^n$.

The global solutions of (1.1) were characterized in classical papers of Gay [1981], Sorensen [1982], and Moré and Sorensen [1983]. Here we are interested in the characterization of local-nonglobal solutions of (1.1) and (1.2).

Our interest in local-nonglobal minimizers comes from the following problem:

$$\begin{aligned} &\text{Minimize } q(s) \\ &\text{s.t. } h(s) \leq 0, \\ &\quad\ ||s|| \leq \Delta, \end{aligned} \tag{1.4}$$

where $h : \mathbb{R}^n \to \mathbb{R}^m$ is continuous and q is given by (1.3). If s^* is a global solution of (1.4) and $h(s^*) < 0$, then s^* is a local (not necessarily global) solution of (1.1). Therefore, local minimizers of (1.1) are natural candidates to be global minimizers of (1.4). Celis, Dennis, and Tapia [1984], Powell and Yuan [1991], and Yuan [1988] proposed trust-region algorithms for solving nonlinear programming problems, where problems like (1.4) are solved at each iteration. The resolution of these subproblems in the two-dimensional case provided the first motivation for the present research. (See Williamson [1990] and Dennis, Martínez, Tapia, and Williamson [1990].) In Williamson's thesis, the problem of finding the global minimizer of a quadratic on the

* Received by the editors September 25, 1991; accepted for publication (in revised form) November 12, 1992. This work was supported by FAPESP grant 90/3724/6, CNPq, FINEP and FAEP-UNICAMP.

† Department of Applied Mathematics, IMECC–UNICAMP, State University of Campinas, CP 6065, 13081 Campinas SP, Brazil (MARTINEZ@CCVAX.UNICAMP.BR).

intersection of a two-dimensional ball and the level set of a convex quadratic is solved. Analytical expressions for all the solutions are derived in a number of special cases, and necessary and sufficient conditions are given for the existence of a nonglobal solution for the general case of the two-dimensional unconstrained trust-region problem.

A very important particular case of (1.4) arises when $h(s) = ||x + s|| - R$. In this case (1.4) is the problem of minimizing an arbitrary quadratic function on the intersection of two Euclidean balls. Heinkenschloss [1990] considered problems of this type when dealing with subproblems related to trust-region strategies for solving

$$\begin{aligned} &\text{Minimize } \tfrac{1}{2}||F(x)||^2 \\ &\text{s.t. } ||x|| \leq R. \end{aligned} \tag{1.5}$$

Problem (1.5) appears in parameter identification problems. In the approach of Heinkenschloss, the Gauss–Newton method is used, so the problem (1.4) is convex and hence, local minimizers are necessarily global. More generally, Martínez and Santos [1991] and Santos [1991] proposed a trust-region strategy for solving

$$\begin{aligned} &\text{Minimize } f(x) \\ &\text{s.t. } ||x|| \leq R. \end{aligned}$$

At each iteration of their algorithm the following subproblem must be solved

$$\begin{aligned} &\text{Minimize } \tfrac{1}{2}s^T G_k s + \nabla f(x_k)^T s \equiv q_k(s) \\ &\text{s.t. } ||x_k + s|| \leq R, \\ &\quad ||s|| \leq \Delta_k. \end{aligned} \tag{1.6}$$

Since the Hessian approximation G_k is not necessarily positive semidefinite, local minimizers of $q_k(s)$ on any of the two balls may be global solutions of (1.6). Nonconvex subproblems may also appear in the Heinkenschloss approach for parameter identification if, instead of the Gauss–Newton method, secant procedures are used to update the Hessian approximation. (See Dennis, Gay, and Welsch [1981], Dennis, Martínez, and Tapia [1988], Martínez [1990], and Engels and Martínez [1991].)

This paper is organized as follows. In §2 we review the characterization of global minimizers and we prove some properties of local minimizers of (1.1) and (1.2). In §3 we give our characterization of local-nonglobal minimizers. An algorithm for computing the local-nonglobal minimizer is given in §4. Some conclusions are stated in §5.

2. Characterization of local and global minimizers. In this section we characterize global minimizers and local minimizers of (1.1) and (1.2). This is in preparation for the characterization of local-nonglobal minimizers, which is given in §3.

The characterization of global minimizers of (1.1) was given independently by Gay [1981] and Sorensen [1982]. Their results were motivated by an earlier work of Goldfeld, Quandt, and Trotter [1966]. In this section we formulate this characterization following Sorensen [1982], who gave very short proofs that are applicable to (1.2).

The following lemma states properties of local minimizers of (1.1) and (1.2) that are proved using classical optimization theory (see, for example, Luenberger [1984]).

LEMMA 2.1. (i) *Assume that s^* is a local minimizer of* (1.2). *Then there exists a unique $\mu^* \in \mathbb{R}$ such that*

$$(G + \mu^* I)s^* + g = 0 \tag{2.1}$$

and

$$w^T(G + \mu^* I)w \geq 0 \tag{2.2}$$

for all $w \in \mathbb{R}^n$ such that $w^T s^* = 0$.

(ii) *Assume that s^* is a local minimizer of* (1.1). *If $||s^*|| = \Delta$, there exists $\mu^* \geq 0$ such that* (2.1) *and* (2.2) *hold. If $||s^*|| < \Delta$ then s^* is a global minimizer and G is positive semidefinite.*

(iii) *Assume that $||s^*|| = \Delta$, $\mu^* \in \mathbb{R}$ satisfies* (2.1), *and*

$$w^T(G + \mu^* I)w > 0 \tag{2.3}$$

for all $w \neq 0$ such that $w^T s^* = 0$. Then s^* is a strict local minimizer of (1.2).

(iv) *Assume that $||s^*|| = \Delta$ and that $\mu^* > 0$ satisfies* (2.1) *and* (2.3). *Then s^* is a strict local minimizer of* (1.1).

Proof. These properties are straightforward applications of the optimality conditions for constrained optimization (see Luenberger [1984], Fletcher [1987], etc.). □

The following two lemmas are essentially Lemmas 2.4 and 2.8 of Sorensen [1982].

LEMMA 2.2. *If s^* is a global minimizer of* (1.2), *there exists $\mu^* \in \mathbb{R}$ such that* (2.1) *holds and $G+\mu^* I$ is positive semidefinite. If, in addition, s^* is a global minimizer of* (1.1) *then $\mu^* \geq 0$.*

If s^* is a global minimizer of (1.1) *and $||s^*|| < \Delta$ then* (2.1) *holds with $\mu^* = 0$ and G is positive semidefinite.*

Proof. See the proof of Lemma 2.4 of Sorensen [1982]. □

LEMMA 2.3. *Assume that $s^* \in \mathbb{R}^n$ and $\mu^* \in \mathbb{R}$ satisfy* (2.1) *and that $G + \mu^* I$ is positive semidefinite.*

(i) *If $||s^*|| < \Delta$ and $\mu^* = 0$ then s^* is a global minimizer of* (1.1).

(ii) *If $||s^*|| = \Delta$ then s^* is a global minimizer of* (1.2).

(iii) *If $||s^*|| = \Delta$ and $\mu^* \geq 0$ then s^* is a global minimizer of* (1.1).

Proof. See the proof of Lemma 2.8 of Sorensen [1982]. □

From now on, we shall consider the spectral decomposition of the symmetric matrix G. (See Golub and Van Loan [1989].) Namely, let $Q \equiv (v_1, \ldots, v_n) \in \mathbb{R}^{n \times n}$ be an orthogonal matrix ($QQ^T = I$), and let $D = \text{Diag}(\lambda_1, \ldots, \lambda_n)$, $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$, be such that

$$G = QDQ^T. \tag{2.4}$$

The following lemma provides a first characterization of local minimizers using (2.4).

LEMMA 2.4. *If s^* is a local minimizer of* (1.1) *or* (1.2), *then* (2.1) *holds with $\mu^* \geq -\lambda_2$.*

Proof. If s^* is a local minimizer of (1.1) and $||s^*|| < \Delta$ then, by Lemma 2.1, G is positive semidefinite and so, $-\lambda_i \leq 0$ for all $i = 1, \ldots, n$. In particular, $-\lambda_2 \leq 0$. But (2.1) holds for $\mu^* = 0$ in this case, so the desired result is proved. If s^* is a local minimizer of (1.1) and $||s^*|| = \Delta$, then s^* is also a local minimizer of (1.2). In this

case, (2.1) and (2.2) hold by Lemma 2.1. Suppose that $\mu^* < -\lambda_2$. By (2.4), we have

$$G + \mu^* I = Q(D + \mu^* I)Q^T = Q \begin{bmatrix} \lambda_1 + \mu^* & & & \bigcirc \\ & \lambda_2 + \mu^* & & \\ & & \ddots & \\ \bigcirc & & & \lambda_n + \mu^* \end{bmatrix} Q^T, \tag{2.5}$$

where $Q = (v_1, \ldots, v_n)$.

Consider the plane P spanned by v_1 and v_2. If $v \in P$ then $v = \alpha v_1 + \beta v_2, \alpha, \beta \in \mathbb{R}$. So,

$$\begin{aligned} v^T(G + \mu^* I)v &= (\alpha v_1 + \beta v_2)^T(G + \mu^* I)(\alpha v_1 + \beta v_2) \\ &= \alpha^2 v_1^T(G + \mu^* I)v_1 + 2\alpha\beta v_1^T(G + \mu^* I)v_2 + \beta^2 v_2^T(G + \mu^* I)v_2. \end{aligned} \tag{2.6}$$

But

$$\begin{aligned} v_i^T(G + \mu^* I)v_j &= v_i^T(v_1, \ldots, v_n) \begin{bmatrix} \lambda_1 + \mu^* & & & \bigcirc \\ & \lambda_2 + \mu^* & & \\ & & \ddots & \\ \bigcirc & & & \lambda_n + \mu^* \end{bmatrix} \begin{pmatrix} v_1^T \\ \vdots \\ v_n^T \end{pmatrix} v_j \\ &= e_i^T \begin{bmatrix} \lambda_1 + \mu^* & & & \bigcirc \\ & \lambda_2 + \mu^* & & \\ & & \ddots & \\ \bigcirc & & & \lambda_n + \mu^* \end{bmatrix} e_j, \end{aligned} \tag{2.7}$$

for $i, j = 1, \ldots, n$, where $\{e_1, \ldots, e_n\}$ denotes the canonical basis on $\mathbb{R}^n$. By (2.7),

$$v_1^T(G + \mu^* I)v_1 = \lambda_1 + \mu^* \leq \lambda_2 + \mu^* < 0, \tag{2.8}$$

$$v_2^T(G + \mu^* I)v_2 = \lambda_2 + \mu^* < 0, \tag{2.9}$$

and

$$v_1^T(G + \mu^* I)v_2 = 0. \tag{2.10}$$

By (2.6) and (2.8)–(2.10),

$$v^T(G + \mu^* I)v < 0 \tag{2.11}$$

for all $v \in P$ such that $v \neq 0$.

Now, the dimension of P is 2 and the dimension of the orthogonal subspace to s^* is $n - 1$. Therefore, there exists $w \neq 0$ such that $w \in P$ and $w^T s^* = 0$.

By (2.11),

$$w^T(G + \mu^* I)w < 0.$$

This contradicts (2.2). Therefore, the assumption $\mu^* < -\lambda_2$ is false and hence, the desired result is proved. □

3. Characterization of local-nonglobal minimizers. In §2 we saw that a local minimizer of (1.2) must satisfy (2.1) with $\mu^* \geq -\lambda_2$. The first lemma in this section shows that $\mu^* \in [-\lambda_2, -\lambda_1)$ when s^* is a local-nonglobal minimizer. Later, we will prove that, in fact, $\mu^* \in (-\lambda_2, -\lambda_1)$ in that case.

LEMMA 3.1. *If s^* is a local-nonglobal minimizer of* (1.2) *or* (1.1) *then* (2.1) *holds with $\mu^* \in [-\lambda_2, -\lambda_1)$.*

Proof. Assume that s^* is a local-nonglobal minimizer of (1.2). By Lemma 2.4, (2.1) holds with $\mu^* \geq -\lambda_2$. But, since s^* is not a global minimizer of (1.2), we have, by Lemma 2.3 (ii), that $\mu^* < -\lambda_1$.

Now assume that s^* is a local-nonglobal minimizer of (1.1). Again, (2.1) holds with $\mu^* \geq -\lambda_2$. Suppose that $\mu^* \geq -\lambda_1$. If $-\lambda_1 \geq 0$, we have by Lemma 2.3 (iii) that s^* is a global minimizer of (1.1). If $\lambda_1 \geq 0$ then G is positive semidefinite, the problem (1.1) is convex, and so there are no local-nonglobal minimizers. This completes the proof. □

Unlike global minimizers, local-nonglobal minimizers of (1.1) or (1.2) do not always exist. We can see that this is the case if $\lambda_1 = \lambda_2$. The following lemma describes another situation where there are no local-nonglobal minimizers.

LEMMA 3.2. *If g is orthogonal to some eigenvector associated with λ_1, then there are no local-nonglobal minimizers of* (1.1) *or* (1.2).

Proof. If $\lambda_1 = \lambda_2$, the result is immediatly true by Lemma 2.1. So, we only need to consider the case when $\lambda_1 < \lambda_2$. If s^* is a local-nonglobal minimizer, there exists $\mu^* \in [-\lambda_2, -\lambda_1)$ such that (2.1) holds. Suppose that $\mu^* \in (-\lambda_2, -\lambda_1)$. In this case, $G + \mu^* I$ is nonsingular, so

$$s^* = -(G + \mu^* I)^{-1} g. \tag{3.1}$$

So, by (2.4) and (3.1),

$$s^* = -(v_1, \ldots, v_n)(D + \mu^* I)^{-1} \begin{pmatrix} v_1^T g \\ \vdots \\ v_n^T g \end{pmatrix}.$$

Now, since $\lambda_1 \neq \lambda_2$, the only eigenvector associated with λ_1 in $\{v_1, \ldots, v_n\}$ is v_1. So, by the hypothesis,

$$v_1^T g = 0. \tag{3.2}$$

Therefore,

$$s^* = -\sum_{i=2}^{n} \frac{c_i}{\lambda_i + \mu^*} v_i, \tag{3.3}$$

where $c_i = v_i^T g, i = 1, \ldots, n$.

By (3.3) and the orthogonality of Q, we have that

$$v_1^T s^* = 0.$$

But

$$v_1^T (G + \mu^* I) v_1 = e_1^T (D + \mu^* I) e_1 = \lambda_1 + \mu^* < 0. \tag{3.4}$$

So, the necessary condition (2.2) is not satisfied. Therefore, s^* cannot be a local minimizer of (1.2), and so s^* is not a local minimizer of (1.1) either.

Now, suppose that s^* is a local-nonglobal minimizer of (1.2) and that (2.1) holds with $\mu^* = -\lambda_2$. Then s^* is a particular solution of the consistent linear system

$$(G - \lambda_2 I)s + g = 0.$$

Therefore,

$$s^* = -(G - \lambda_2 I)^\dagger g + \sum_{\lambda_j = \lambda_2} \alpha_j v_j, \tag{3.5}$$

where $A^\dagger$ denotes the Moore–Penrose pseudoinverse of A.

Now, using (3.2), we have that

$$(G - \lambda_2 I)^\dagger g = -\sum_{\substack{j=3 \\ \lambda_j \neq \lambda_2}}^{n} \frac{c_j}{\lambda_j - \lambda_2} v_j. \tag{3.6}$$

By (3.5) and (3.6) we deduce that $v_1^T s^* = 0$. But, in this case, as in (3.4), we have

$$v_1^T(G + \mu^* I)v_1 = v_1^T(G - \lambda_2 I)v_1 = \lambda_1 - \lambda_2 < 0.$$

So, the necessary condition (2.2) is not satisfied and thus s^* is not a local minimizer of (1.2). This completes the proof. □

In the following lemma we prove that the multiplier μ^* must be strictly greater than $-\lambda_2$ when s^* is a local-nonglobal minimizer of (1.1) or (1.2).

LEMMA 3.3. *If s^* is a local-nonglobal minimizer of* (1.2) *or* (1.1), *then* (2.1) *holds with $\mu^* \in (-\lambda_2, -\lambda_1)$.*

Proof. As in Lemma 3.2, we only need to consider the case when $\lambda_1 < \lambda_2$. Assume that s^* is a local-nonglobal minimizer of (1.2). By Lemma 2.4, (2.1) holds with $\mu^* \geq -\lambda_2$. Since s^* is not a global minimizer then $\mu^* \in [-\lambda_2, -\lambda_1)$. Suppose that $\mu^* = -\lambda_2$. Define the plane P as in Lemma 2.4. Since $\lambda_1 < \lambda_2$ we have, as in (2.8), that

$$v_1^T(G + \mu^* I)v_1 = v_1^T(G - \lambda_2 I)v_1 = \lambda_1 - \lambda_2 < 0,$$

but

$$v_2^T(G + \mu^* I)v_2 = v_2^T(G - \lambda_2 I)v_2 = \lambda_2 - \lambda_2 = 0.$$

So, since (2.10) obviously holds, we have that

$$v^T(G + \mu^* I)v = v^T(G - \lambda_2 I)v < 0$$

for all $v \in P$ such that $v \notin \mathrm{Span}(\{v_2\})$. Let $v \neq 0$ be such that $v^T s^* = 0$ and $v \in P$. By (2.2), $v^T(G - \lambda_2 I)v \geq 0$, so $v \in \mathrm{Span}(\{v_2\})$. Therefore,

$$v_2^T s^* = 0. \tag{3.7}$$

Now, since s^* is a local minimizer of (1.2), it follows that $w^* \equiv (v_1^T s^*, v_2^T s^*, \ldots, v_n^T s^*)^T$ is a local minimizer of

$$\begin{aligned} &\text{Minimize} \quad \frac{1}{2}\sum_{i=1}^{n}\lambda_i w_i^2 + c^T w \\ &\text{s.t.} \quad\quad ||w|| = \Delta, \end{aligned}$$

where $c = Q^T g = (v_1^T g, \ldots, v_n^T g)^T$. Moreover, by (2.1), we have that

$$(\lambda_i - \lambda_2)w_i^* + c_i = 0, \tag{3.8}$$

$i = 1, \ldots, n$. So, $c_2 = 0$ and, by (3.7) and (3.8),

$$\begin{aligned} w_1^* &= -c_1/(\lambda_1 - \lambda_2), \\ w_2^* &= 0. \end{aligned} \tag{3.9}$$

By Lemma 3.2, $c_1 = v_1^T g \neq 0$. So, $w_1^* \neq 0$. Assume, without loss of generality, that $w_1^* > 0$. Therefore, w^* is a local minimizer of

$$\begin{aligned} &\text{Minimize} \quad \frac{1}{2}\sum_{i=1}^{n}\lambda_i w_i^2 + c_1 w_1 + \sum_{i=3}^{n} c_i w_i \\ &\text{s.t.} \quad\quad w_1 = \sqrt{\Delta^2 - \sum_{i=2}^{n} w_i^2}. \end{aligned} \tag{3.10}$$

Hence $(w_2^*, \ldots, w_n^*)^T$ is an unconstrained local minimizer of the function

$$\begin{aligned} F(w_2, \ldots, w_n) &= \frac{1}{2}\lambda_1\left(\Delta^2 - \sum_{i=2}^{n} w_i^2\right) + \sum_{i=2}^{n}\lambda_i w_i^2 \\ &\quad + c_1\sqrt{\Delta^2 - \sum_{i=2}^{n} w_i^2} + \sum_{i=3}^{n} c_i w_i \\ &= \frac{1}{2}\sum_{i=2}^{n}(\lambda_i - \lambda_1)w_i^2 + c_1\sqrt{\Delta^2 - \sum_{i=2}^{n} w_i^2} + \sum_{i=3}^{n} c_i w_i + K, \end{aligned}$$

where $K = \lambda_1\Delta^2/2$.

So, w_2^* is an unconstrained local minimizer of the scalar-valued function ψ_1 given by

$$\psi_1(y) = \frac{1}{2}(\lambda_2 - \lambda_1)y^2 + c_1\sqrt{\Delta^2 - y^2 - \sum_{i=3}^{n}(w_i^*)^2} + \sum_{i=3}^{n} c_i w_i^* + \frac{1}{2}\sum_{i=3}^{n}(\lambda_i - \lambda_1)(w_i^*)^2 + K.$$

Therefore, 0 is a local minimizer of

$$\psi(y) = \frac{y^2}{2} + \frac{c_1}{\lambda_2 - \lambda_1}\sqrt{\Delta^2 - y^2 - \sum_{i=3}^{n}(w_i^*)^2}.$$

By (3.9),

$$\psi(y) = \frac{y^2}{2} + w_1^* \sqrt{\Delta^2 - y^2 - \sum_{i=3}^{n} (w_i^*)^2}.$$

So, by (3.9) and (3.10),

$$\psi(y) = \frac{y^2}{2} + w_1^* \sqrt{(w_1^*)^2 - y^2}.$$

Writing $b \equiv b(y) \equiv (w_1^*)^2 - y^2$, we calculate

$$\psi'(y) = y - w_1^* y / \sqrt{b},$$

$$\psi''(y) = 1 - \frac{w_1^*}{\sqrt{b}} - \frac{w_1^* y^2}{b^{3/2}},$$

$$\psi'''(y) = \frac{-3w_1^* y}{b^{3/2}} - \frac{3y^3 w_1^*}{b^{5/2}},$$

$$\psi''''(y) = \frac{-3w_1^*}{b^{3/2}} - \frac{18w_1^* y^2}{b^{5/2}} - \frac{15y^4 w_1^*}{b^{7/2}}.$$

Therefore, $\psi'(0) = \psi''(0) = \psi'''(0) = 0$, but $\psi''''(0) < 0$. This contradicts the assumption that 0 is a local minimizer of ψ. The contradiction comes from the initial assumption that s^* is a local-nonglobal minimizer of (1.2) with $\mu^* = -\lambda_2$.

If s^* is a local-nonglobal minimizer of (1.1) we necessarily have that $||s^*|| = \Delta$, and that (2.1) holds with $\mu^* \in [-\lambda_2, -\lambda_1)$. If $\mu^* = -\lambda_2$, the arguments above show that s^* cannot be a local minimizer of (1.2). So, s^* cannot be a local minimizer of (1.1). This completes the proof. □

Now we are prepared to state the main result of this section. By the previous lemmas we know that if s^* is a local-nonglobal minimizer of (1.1) or (1.2), then $||s^*|| = \Delta$, $(G + \mu^* I)s^* = -g$, and $\mu^* \in (-\lambda_2, -\lambda_1)$. Since $G + \mu I$ is nonsingular for all $\mu \in (-\lambda_2, -\lambda_1)$, we have that

$$s^* = -(G + \mu^* I)^{-1} g$$

and

$$||(G + \mu^* I)^{-1} g|| = \Delta.$$

Let us define

$$\varphi(\mu) = ||(G + \mu I)^{-1} g||^2 \tag{3.11}$$

for all $\mu \in (-\lambda_2, -\lambda_1)$.

By (2.4), we have that

$$\varphi(\mu) = \sum_{i=1}^{n} \frac{c_i^2}{(\lambda_i + \mu)^2},$$

$$\varphi'(\mu) = -2\sum_{i=1}^{n} \frac{c_i^2}{(\lambda_i + \mu)^3}, \quad \text{and} \tag{3.12}$$

$$\varphi''(\mu) = 6\sum_{i=1}^{n} \frac{c_i^2}{(\lambda_i + \mu)^4},$$

for all $\mu \in (-\lambda_2, -\lambda_1)$, where $c = Q^T g$.

If a local-nonglobal minimizer exists, we must have $c_1 \neq 0$ and $\lambda_1 < \lambda_2$. Therefore, by (3.12), φ is a strictly convex function on $(-\lambda_2, -\lambda_1)$ in this case. Moreover,

$$\lim_{\mu \to -\lambda_1} \varphi(\mu) = \infty. \tag{3.13}$$

Therefore, the equation $\varphi(\mu) = \Delta^2$, which characterizes local minimizers, has at most two roots in $(-\lambda_2, -\lambda_1)$. The following theorem shows that only the largest root can correspond to a local minimizer of (1.1) or (1.2). A consequence of this fact is that a local-nonglobal minimizer is necessarily unique.

THEOREM 3.1. (i) *If s^* is a local-nonglobal minimizer of* (1.1) *or* (1.2), *then* (2.1) *holds with $\mu^* \in (-\lambda_2, -\lambda_1)$ and $\varphi'(\mu^*) \geq 0$. If s^* is a local-nonglobal minimizer of* (1.1) *then $\mu^* \geq 0$.*

(ii) *There exists at most one local-nonglobal minimizer of* (1.2) *or* (1.1).

(iii) *If $\|s^*\| = \Delta$,* (2.1) *holds for some $\mu^* \in (-\lambda_2, -\lambda_1)$ and $\varphi'(\mu^*) > 0$, then s^* is a strict local minimizer of* (1.2). *If, in addition, $\mu^* > 0$, s^* is also a strict local minimizer of* (1.1).

Proof. Let us prove (i).

Let s^* be a local-nonglobal minimizer of (1.2). We know, from the previous lemmas, that (2.1) holds for some $\mu^* \in (-\lambda_2, -\lambda_1)$.

Then, $G + \mu^* I$ is nonsingular and, by (2.4),

$$s^* = -(G + \mu^* I)^{-1} g = -\sum_{i=1}^{n} \frac{c_i}{\lambda_i + \mu^*} v_i. \tag{3.14}$$

Define $W(\mu) \in \mathbb{R}^{n \times (n-1)}$ by

$$W(\mu) = \left(\frac{c_2}{\lambda_2 + \mu} v_1 - \frac{c_1}{\lambda_1 + \mu} v_2,\ \frac{c_3}{\lambda_3 + \mu} v_1 - \frac{c_1}{\lambda_1 + \mu} v_3, \ldots, \frac{c_n}{\lambda_n + \mu} v_1 - \frac{c_1}{\lambda_1 + \mu} v_n \right)$$

$$= (v_1, \ldots, v_n) \begin{bmatrix} \frac{c_2}{\lambda_2 + \mu} & \frac{c_3}{\lambda_3 + \mu} & \cdots & \frac{c_n}{\lambda_n + \mu} \\ \frac{-c_1}{\lambda_1 + \mu} & 0 & \cdots & 0 \\ 0 & \frac{-c_1}{\lambda_1 + \mu} & \cdots & \vdots \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & & \frac{-c_1}{\lambda_1 + \mu} \end{bmatrix} \tag{3.15}$$

for all $\mu \in (-\lambda_2, -\lambda_1)$.

By Lemma 3.2, $c_1 = v_1^T g \neq 0$, so

$$\text{rank } (W(\mu)) = n - 1 \tag{3.16}$$

for all $\mu \in (-\lambda_2, -\lambda_1)$.

Moreover,

$$\begin{aligned}(s^*)^T W(\mu^*) e_j &= \left[-\sum_{i=1}^{n} \frac{c_i v_i}{(\lambda_i + \mu^*)}\right]^T \left(\frac{c_{j+1}}{\lambda_{j+1} + \mu^*} v_1 - \frac{c_1}{\lambda_1 + \mu^*} v_{j+1}\right) \\ &= \frac{-c_1 c_{j+1}}{(\lambda_1 + \mu^*)(\lambda_{j+1} + \mu^*)} + \frac{c_1 c_{j+1}}{(\lambda_{j+1} + \mu^*)(\lambda_1 + \mu^*)} = 0 \end{aligned} \tag{3.17}$$

for $j = 1, \ldots, n-1$.

By (3.16) and (3.17), the columns of $W(\mu^*)$ form a basis of the hyperplane orthogonal to s^*.

Define

$$B(\mu) = W(\mu)^T (G + \mu I) W(\mu) \tag{3.18}$$

for all $\mu \in (-\lambda_2, -\lambda_1)$.

Therefore,

$$B(\mu) = \begin{bmatrix} \frac{c_2^2(\lambda_1+\mu)}{(\lambda_2+\mu)^2} + \frac{c_1^2(\lambda_2+\mu)}{(\lambda_1+\mu)^2} & \frac{c_2 c_3(\lambda_1+\mu)}{(\lambda_2+\mu)(\lambda_3+\mu)} & \cdots & \frac{c_2 c_n(\lambda_1+\mu)}{(\lambda_2+\mu)(\lambda_n+\mu)} \\ \frac{c_2 c_3(\lambda_1+\mu)}{(\lambda_2+\mu)(\lambda_3+\mu)} & \frac{c_3^2(\lambda_1+\mu)}{(\lambda_3+\mu)^2} + \frac{c_1^2(\lambda_3+\mu)}{(\lambda_1+\mu)^2} & \cdots & \frac{c_3 c_n(\lambda_1+\mu)}{(\lambda_3+\mu)(\lambda_n+\mu)} \\ \vdots & \vdots & & \vdots \\ \frac{c_2 c_n(\lambda_1+\mu)}{(\lambda_2+\mu)(\lambda_n+\mu)} & \frac{c_3 c_n(\lambda_1+\mu)}{(\lambda_3+\mu)(\lambda_n+\mu)} & \cdots & \frac{c_n^2(\lambda_1+\mu)}{(\lambda_n+\mu)^2} + \frac{c_1^2(\lambda_n+\mu)}{(\lambda_1+\mu)^2} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{c_1^2(\lambda_2+\mu)}{(\lambda_1+\mu)^2} & & & 0 \\ & \frac{c_1^2(\lambda_3+\mu)}{(\lambda_1+\mu)^2} & & \\ & & \ddots & \\ 0 & & & \frac{c_1^2(\lambda_n+\mu)}{(\lambda_1+\mu)^2} \end{bmatrix} + \sigma u u^T, \tag{3.19}$$

where

$$u = \begin{bmatrix} \frac{c_2}{\lambda_2+\mu} \\ \vdots \\ \frac{c_n}{\lambda_n+\mu} \end{bmatrix}, \qquad \sigma = \lambda_1 + \mu. \tag{3.20}$$

Hence

$$B(\mu) = \widehat{B}(\mu)\,(I + \sigma \widehat{B}(\mu)^{-1} u u^T), \tag{3.21}$$

where

$$\widehat{B}(\mu) = \begin{bmatrix} \dfrac{c_1^2(\lambda_2+\mu)}{(\lambda_1+\mu)^2} & & 0 \\ & \ddots & \\ 0 & & \dfrac{c_1^2(\lambda_n+\mu)}{(\lambda_1+\mu)^2} \end{bmatrix}. \tag{3.22}$$

Thus (see Golub and Van Loan [1989]),

$$\det(B(\mu)) = \det(\widehat{B}(\mu))\,(1+\sigma u^T[\widehat{B}(\mu)]^{-1}u). \tag{3.23}$$

Now,

$$\det(\widehat{B}(\mu)) = \frac{c_1^{2n-2}(\lambda_2+\mu)\cdots(\lambda_n+\mu)}{(\lambda_1+\mu)^{2n-2}}, \tag{3.24}$$

and

$$\begin{aligned} 1+\sigma u^T[\widehat{B}(\mu)]^{-1}u &= 1+(\lambda_1+\mu)\left(\frac{c_2}{\lambda_2+\mu},\ldots,\frac{c_n}{\lambda_n+\mu}\right)\begin{pmatrix} \dfrac{(\lambda_1+\mu)^2c_2}{c_1^2(\lambda_2+\mu)^2} \\ \vdots \\ \dfrac{(\lambda_1+\mu)^2c_n}{c_1^2(\lambda_n+\mu)^2} \end{pmatrix} \\ &= 1+\frac{c_2^2(\lambda_1+\mu)^3}{c_1^2(\lambda_2+\mu)^3}+\cdots+\frac{c_n^2(\lambda_1+\mu)^3}{c_1^2(\lambda_n+\mu)^3}. \end{aligned} \tag{3.25}$$

Hence, by (3.19)–(3.25),

$$\begin{aligned} \det(B(\mu)) &= \frac{c_1^{2n-2}(\lambda_2+\mu)\cdots(\lambda_n+\mu)(\lambda_1+\mu)^3}{(\lambda_1+\mu)^{2n-2}} \\ &\quad\times\left[\frac{1}{(\lambda_1+\mu)^3}+\frac{c_2^2}{c_1^2(\lambda_2+\mu)^3}+\cdots+\frac{c_n^2}{c_1^2(\lambda_n+\mu)^3}\right] \\ &= \frac{c_1^{2n-4}(\lambda_2+\mu)\cdots(\lambda_n+\mu)}{(\lambda_1+\mu)^{2n-5}}\left[\frac{c_1^2}{(\lambda_1+\mu)^3}+\cdots+\frac{c_n^2}{(\lambda_n+\mu)^3}\right] \\ &= \frac{-c_1^{2n-4}(\lambda_2+\mu)\cdots(\lambda_n+\mu)}{2(\lambda_1+\mu)^{2n-5}}\left[\frac{-2c_1^2}{(\lambda_1+\mu)^3}+\cdots+\frac{-2c_n^2}{(\lambda_n+\mu)^3}\right]. \end{aligned} \tag{3.26}$$

So, by (3.26) and (3.12),

$$\det\ (B(\mu)) = \frac{-c_1^{2n-4}(\lambda_2+\mu)\cdots(\lambda_n+\mu)}{2(\lambda_1+\mu)^{2n-5}}\varphi'(\mu) \tag{3.27}$$

for all $\mu \in (-\lambda_2, -\lambda_1)$. But, since $c_1 \neq 0$, we deduce that

$$\frac{-c_1^{2n-4}(\lambda_2+\mu)\cdots(\lambda_n+\mu)}{2(\lambda_1+\mu)^{2n-5}} > 0 \tag{3.28}$$

for all $\mu \in (-\lambda_2, -\lambda_1)$. Now, by (2.2) and (3.18) we have that $B(\mu^*)$ is positive semidefinite. So, $\det(B(\mu^*)) \geq 0$. Hence, by (3.27) and (3.28),

$$\varphi'(\mu^*) \geq 0.$$

To finish the proof of (i) observe that, by Lemma 2.1, if s^* is a local minimizer of (1.1), we must have $\mu^* \geq 0$.

Part (ii) of the theorem follows from the elementary analysis of the function φ given by (3.12). In fact, it follows from this analysis and (i) that only the largest root of $\varphi(\mu) = \Delta^2$ in $(-\lambda_2, -\lambda_1)$ can define a local-nonglobal minimizer.

Let us now prove part (iii). Assume that s^* satisfies (2.1) for some $\mu^* \in (-\lambda_2, -\lambda_1)$, $||s^*|| = \Delta$, and $\varphi'(\mu^*) > 0$. Since φ' is strictly increasing in $(-\lambda_2, -\lambda_1)$ we have that

$$\varphi'(\mu) > 0 \tag{3.29}$$

for all $\mu \in [\mu^*, -\lambda_1)$.

Suppose that s^* is not a strict local minimizer of (1.2). Define $W(\mu)$ as in (3.15) and $B(\mu)$ as in (3.18). So, (3.27) and (3.28) hold for all $\mu \in [\mu^*, -\lambda_1)$.

Now, by (3.19), all the eigenvalues of $B(\mu)$ are strictly positive if $\mu \in (-\lambda_2, -\lambda_1)$ is close enough to $-\lambda_1$. If s^* is not a strict local minimizer of (1.2) we deduce, as in (i), that $B(\mu^*)$ has some eigenvalue less than or equal to zero. Therefore, there exists $\widetilde{\mu} \in [\mu^*, -\lambda_1)$ such that $B(\widetilde{\mu})$ is singular. So,

$$0 = \det(B(\widetilde{\mu})) = \frac{-c_1^{2n-4}(\lambda_2 + \widetilde{\mu}) \cdots (\lambda_n + \widetilde{\mu})}{2(\lambda_1 + \widetilde{\mu})^{2n-5}} \varphi'(\widetilde{\mu}). \tag{3.30}$$

Equation (3.30) implies that $\varphi'(\widetilde{\mu}) = 0$, contradicting (3.29). Thus (2.1) together with $||s^*|| = \Delta$, $\mu^* \in (-\lambda_2, -\lambda_1)$, and $\varphi'(\mu^*) > 0$ implies that s^* is a strict local-nonglobal minimizer of (1.2). If, in addition, $\mu^* > 0$, s^* is also a strict local minimizer of (1.1), by the optimality conditions of nonlinear programming. (See Luenberger [1984, pp. 316–317].) This completes the proof of the theorem. □

We finish this section proving a lemma that helps to understand the structure of local-nonglobal minimizers.

LEMMA 3.4. *Consider the problem* (1.2), *where* G, D, *and* Q *satisfy* (2.4), $c = Q^T g$, $\lambda_1 < \lambda_2$, *and* $c_1 \neq 0$. *Then there exists* $\Delta_0 > 0$ *such that* (1.2) *admits a local-nonglobal minimizer for all* $\Delta > \Delta_0$. *Calling* $s_G(\Delta)$ *the global minimizer of* (1.2), $s_L(\Delta)$ *the local-nonglobal minimizer of* (1.2) *for* $\Delta > \Delta_0$, *we have*

$$\lim_{\Delta \to \infty} \frac{s_G(\Delta)}{\Delta} = -v_1 \frac{c_1}{|c_1|} \tag{3.31}$$

and

$$\lim_{\Delta \to \infty} \frac{s_L(\Delta)}{\Delta} = v_1 \frac{c_1}{|c_1|}. \tag{3.32}$$

Proof. By (3.12) and (3.13), the equation $\varphi(\mu) = \Delta^2$ has a root $\mu(\Delta) \in (-\lambda_2, -\lambda_1)$ such that $\varphi'(\mu(\Delta)) > 0$, if Δ is large enough. Moreover,

$$\lim_{\Delta \to \infty} \mu(\Delta) = -\lambda_1. \tag{3.33}$$

Now, by (2.1),

$$\begin{aligned} s_L(\Delta) &= -(G + \mu(\Delta)I)^{-1} g \\ &= -Q \begin{bmatrix} 1/(\lambda_1 + \mu(\Delta)) & & \\ & \ddots & \\ & & 1/(\lambda_n + \mu(\Delta)) \end{bmatrix} c \\ &= -\sum_{i=1}^{n} \frac{c_i}{\lambda_i + \mu(\Delta)} v_i. \end{aligned} \tag{3.34}$$

So, since $\lambda_1 + \mu(\Delta) < 0$,

$$1 = \lim_{\Delta\to\infty} \frac{\|s_L(\Delta)\|}{\Delta} = \lim_{\Delta\to\infty} \sqrt{\sum_{i=1}^{n} \left[\frac{c_i/\Delta}{(\lambda_i + \mu(\Delta))}\right]^2}$$

$$= \lim_{\Delta\to\infty} \frac{-|c_1|/\Delta}{(\lambda_1 + \mu(\Delta))}. \tag{3.35}$$

Hence, by (3.34) and (3.35),

$$\lim_{\Delta\to\infty} \frac{s_L(\Delta)}{\Delta} = -\lim_{\Delta\to\infty} \sum_{i=1}^{n} \frac{(c_1/\Delta)v_i}{\lambda_i + \mu(\Delta)} = \lim_{\Delta\to\infty} -\frac{(c_1/\Delta)v_1}{\lambda_1 + \mu(\Delta)}$$

$$= \lim_{\Delta\to\infty} \frac{-(c_1|c_1|/\Delta)v_1}{|c_1|(\lambda_1 + \mu(\Delta))} = \frac{c_1}{|c_1|} v_1.$$

So, (3.32) is proved. In an analogous way, using the fact that $\lambda_1 + \mu^* > 0$ in the case of global minimizers, we prove (3.31). □

4. Computing the local-nonglobal minimizer. By the results in §3 of this paper, the problem of computing the local-nonglobal minimizer of (1.2) reduces to finding the largest root of

$$\varphi(\mu) = \Delta^2 \tag{4.1}$$

for $\mu \in (-\lambda_2, -\lambda_1)$, where $\varphi(\mu)$ is given by (3.11)–(3.12), $\lambda_1 < \lambda_2$, and $c_1 \neq 0$.

If we are interested in local-nonglobal minimizers of (1.1), it is sufficient to discard μ if it is negative.

Assume that $c_2 = c_3 = \cdots = c_l, l \leq n$. If there exists $i \in \{2, \ldots, l\}$ such that $c_i \neq 0$, we have

$$\lim_{\mu\to-\lambda_2} \varphi(\mu) = \infty.$$

On the other hand, if $c_2 = \cdots = c_l = 0$, we have

$$\lim_{\mu\to-\lambda_2} \varphi(\mu) = \frac{c_1^2}{(\lambda_1 - \lambda_2)^2} + \sum_{i=l+1}^{n} \frac{c_i^2}{(\lambda_i - \lambda_2)^2}.$$

In this section we assume that λ_1 and λ_2 are known.

If the complete eigensystem of G is available, $\varphi(\mu), \varphi'(\mu)$, and $\varphi''(\mu)$ may be computed using (3.12). Otherwise, $\varphi(\mu)$ may be computed by (3.11) and

$$\varphi'(\mu) = 2\sum_{i=1}^{n} \Phi_i(\mu)\Phi_i'(\mu), \tag{4.2}$$

$$\varphi''(\mu) = 2\sum_{i=1}^{n} \Phi_i'(\mu)\Phi_i(\mu) + \Phi_i''(\mu)^2, \tag{4.3}$$

where

$$\Phi(\mu) = -(G + \mu I)^{-1} g, \tag{4.4}$$

$$\Phi'(\mu) = (G + \mu I)^{-1}\Phi(\mu) = (G + \mu I)^{-2} g, \tag{4.5}$$

and

$$\Phi''(\mu) = -2(G + \mu I)^{-1}\Phi'(\mu) = -2(G + \mu I)^{-3}g. \tag{4.6}$$

Clearly, given $\mu \in (-\lambda_2, -\lambda_1)$, (4.2)–(4.6) can be computed using the Cholesky factorization of $G + \mu I$.

The equation (4.1) is equivalent to

$$\gamma(\mu) = \frac{1}{\Delta}, \tag{4.7}$$

where

$$\gamma(\mu) = \varphi(\mu)^{-1/2}.$$

So,

$$\gamma'(\mu) = -\tfrac{1}{2}\varphi(\mu)^{-3/2}\varphi'(\mu)$$

and

$$\gamma''(\mu) = \tfrac{3}{4}\varphi(\mu)^{-5/2}\varphi'(\mu) - \tfrac{1}{2}\varphi(\mu)^{-3/2}\varphi''(\mu).$$

The function γ and the equation (4.7) are used by Reinsch [1971], Hebden [1973], Moré [1978], Gay [1981], and Moré and Sorensen [1983] for finding the global minimizers of (1.1).

In this section we introduce three algorithms for finding the local-nonglobal minimizer of (1.2) or (1.1). The first is based on the quadratic approximation of φ and the other two are based on approximations of γ. For $\overline{\mu} \in (-\lambda_2, -\lambda_1)$, we define $S_1(\overline{\mu})$ to be the largest root of

$$\varphi(\overline{\mu}) + \varphi'(\overline{\mu})(\mu - \overline{\mu}) + \tfrac{1}{2}\varphi''(\overline{\mu})(\mu - \overline{\mu})^2 = \Delta^2.$$

If this root does not exist, or if it does not belong to $(-\lambda_2, -\lambda_1)$, we define $S(\overline{\mu}) = \infty$.

Analogously, $S_2(\overline{\mu})$ will be the largest root of

$$\gamma(\overline{\mu}) + \gamma'(\overline{\mu})(\mu - \overline{\mu}) + \frac{1}{2}\gamma''(\overline{\mu})(\mu - \overline{\mu})^2 = \frac{1}{\Delta}$$

if this root exists and belongs to $(-\lambda_2, -\lambda_1)$, and $S_2(\overline{\mu}) = \infty$ otherwise.

Finally, the fact that γ is not always a concave function motivated us to define a third iteration function $S_3(\overline{\mu})$ as the zero of the linearized γ. So,

$$S_3(\overline{\mu}) = \overline{\mu} - \frac{\gamma(\overline{\mu}) - 1/\Delta}{\gamma'(\overline{\mu})}$$

if the root exists and belongs to $(-\lambda_2, -\lambda_1)$, and $S_3(\overline{\mu}) = \infty$ otherwise. This choice of $S_3(\overline{\mu})$ is analogous to the choice of Hebden and Moré for the problem of computing global minimizers.

The three algorithms differ in the iteration function used, but all obey the general scheme given by the following algorithm. This algorithm generates a sequence of approximations μ_k of the multiplier of the local-nonglobal minimizer, together with a

sequence of lower bounds μ_k^L and a sequence of upper bounds μ_k^U. At the beginning of the process, $\mu_0^L = -\lambda_2, \mu_0^U = -\lambda_1$, and $\mu_0 = (-\lambda_2 - \lambda_1)/2$.

ALGORITHM 4.1. Assume that μ_k, μ_k^L, μ_k^U have been computed, $-\lambda_2 \leq \mu_k^L < \mu_k < \mu_k^U \leq -\lambda_1$.

Step 1. If $\varphi(\mu_k) = \Delta^2$ and $\varphi'(\mu_k) \geq 0$, stop.

Step 2. If $\varphi(\mu_k) \leq \Delta^2$ or $\varphi'(\mu_k) \leq 0$, define $\mu_{k+1}^L = \mu_k, \mu_{k+1}^U = \mu_k^U$.
Otherwise, define $\mu_{k+1}^U = \mu_k, \mu_{k+1}^L = \mu_k^L$.

Step 3. If $\varphi(\mu_k) > \Delta^2$ and $\varphi'(\mu_k) = 0$
or
$\varphi(\mu_k) > \Delta^2$ and $\mu_k^N \notin [\mu_{k+1}^L, \mu_{k+1}^U]$
where

$$\mu_k^N = \mu_k - \frac{(\varphi(\mu_k) - \Delta^2)}{\varphi'(\mu_k)},$$

declare that the local-nonglobal minimizer does not exist and stop.

Step 4. Compute $S_i(\mu_k)(i \in \{1, 2, 3\})$.
If $S_i(\mu_k) \in [\mu_{k+1}^L + 0.01(\mu_{k+1}^U - \mu_{k+1}^L), \mu_{k+1}^L + 0.99(\mu_{k+1}^U - \mu_{k+1}^L)]$, define $\mu_{k+1} = S_i(\mu_k)$.
Otherwise, define $\mu_{k+1} = (\mu_{k+1}^L + \mu_{k+1}^U)/2$.

Step 5. $k \leftarrow k + 1$. Go to Step 1.

The global convergence behavior of Algorithm 4.1 is described by the following theorem.

THEOREM 4.1. *Consider the sequence generated by Algorithm* 4.1.

(i) *If there exists* $\mu^* \in (-\lambda_2, -\lambda_1)$ *such that* $\varphi(\mu^*) = \Delta^2$ *and* $\varphi'(\mu^*) \geq 0$, *then* $\mu^* \in [\mu_L^k, \mu_U^k]$ *for all* $k = 0, 1, 2, \ldots$.

(ii) *If the algorithm stops at Step* 3, *then there is no solution* μ^* *such that* $\varphi(\mu^*) = \Delta^2$ *and* $\varphi'(\mu^*) \geq 0$.

(iii) *If the algorithm does not stop in a finite number of iterations, then there exists* $\mu^* \in [-\lambda_2, -\lambda_1)$ *such that*

$$\lim_{k\to\infty} \mu_k = \lim_{k\to\infty} \mu_k^L = \lim_{k\to\infty} \mu_k^U = \mu^*.$$

(iv) *In the case* (iii), *if* $\mu^* = -\lambda_2$, *there does not exist* $\mu \in (-\lambda_2, -\lambda_1)$ *such that* $\varphi(\mu) = \Delta^2$ *and* $\varphi'(\mu) \geq 0$.

(v) *In the case* (iii), *if* $\mu^* \in (-\lambda_2, -\lambda_1)$ *then* $\varphi(\mu^*) = \Delta^2$ *and* $\varphi'(\mu^*) \geq 0$.

Proof. The part (i) of the theorem follows by induction using elementary properties of φ.

Let us prove (ii). Since φ is convex, we have that

$$\varphi(\mu) \geq \varphi(\mu_k) + \varphi'(\mu_k)(\mu - \mu_k)$$

for all $\mu \in (-\lambda_2, -\lambda_1)$.

But, if the conditions of Step 3 hold, we have

$$\varphi(\mu_k) + \varphi'(\mu_k)(\mu - \mu_k) > \Delta^2$$

for all $\mu \in [\mu_L^{k+1}, \mu_U^{k+1}]$. Therefore,

$$\varphi(\mu) > \Delta^2$$

for all $\mu \in [\mu_L^{k+1}, \mu_U^{k+1}]$. By part (i) of the theorem, this implies that the solution does not exist.

Let us prove (iii). By the choice of Step 4, we have

$$\mu_{k+1} \in [\mu_{k+1}^L + 0.01(\mu_{k+1}^U - \mu_{k+1}^L), \mu_{k+1}^L + 0.99(\mu_{k+1}^U - \mu_{k+1}^L)]$$

for all $k = 0, 1, 2, \ldots$.

Therefore, by the definition of μ_{k+1}^L and μ_{k+1}^U at Step 2, we have

$$\mu_{k+1}^U - \mu_{k+1}^L \leq 0.99(\mu_k^U - \mu_k^L)$$

for all $k = 0, 1, 2, \ldots$. This implies that there exists $\mu^* \in [-\lambda_2, -\lambda_1]$ such that

$$\lim_{k\to\infty} \mu_k^L = \lim_{k\to\infty} \mu_k^U = \lim_{k\to\infty} \mu_k = \mu^*.$$

Suppose that $\mu^* = -\lambda_1$. In this case, μ_k^L must have been changed an infinite number of times at Step 2. So $\varphi(\mu_k) \leq \Delta^2$ or $\varphi'(\mu_k) \leq 0$ and $\lim \mu_k = -\lambda_1$ for some subsequence. This contradicts the fact that

$$\lim_{\mu\to-\lambda_1} \varphi(\mu) = \lim_{\mu\to-\lambda_1} \varphi'(\mu) = \infty.$$

Therefore, (iii) is proved.

Let us prove (iv). In this case, $\lim_{k\to\infty} \mu_k^U = -\lambda_2$. So, by the choice of Step 2, there exists an infinite subset $K_1 \subset \mathbb{N}$ such that

$$\lim_{k\in K_1} \mu_k = -\lambda_2, \quad \mu_k > -\lambda_2, \quad \varphi(\mu_k) > \Delta^2 \quad \text{and} \quad \varphi'(\mu_k) > 0$$

for all $k \in K_1$. By the convexity of φ, this implies that $\varphi'(\mu) > 0$ for all $\mu \in (-\lambda_2, -\lambda_1)$ and $\varphi(\mu) > 0$ for all $\mu \in (-\lambda_2, -\lambda_1)$. So, (iv) is proved.

Finally, let us prove (v). Since $\mu^* = \lim_{k\to\infty} \mu_k^U < -\lambda_1$, we have, by Step 2, that

$$\varphi(\mu^*) \geq \Delta^2 \quad \text{and} \quad \varphi'(\mu^*) \geq 0. \tag{4.8}$$

Suppose that $\varphi(\mu^*) > \Delta^2$. Since $\lim_{k\to\infty} \mu_k^L = \mu^*$, we have, by Step 2, that $\varphi'(\mu^*) \leq 0$. Then, by (4.8), $\varphi'(\mu^*) = 0$. But, in this case, we would have $\mu_k^N \notin [-\lambda_2, -\lambda_1]$ for k large enough and the algorithm would have stopped at Step 3. So, $\varphi(\mu^*) = \Delta^2$ and the theorem is proved. □

We performed some numerical experiments using Algorithm 4.1 and the three different iteration functions S_1, S_2, and S_3. The best results were obtained using S_1. A typical example is shown in Table 1. In this experiment $n = 10$, $\lambda_i = -n - i - 1$, $c_i = n + i + 1$, $i = 1, \ldots, n$, and $\Delta = 30$. We report the quantity $e_k = \sqrt{\varphi(\mu_k)} - \Delta$ at each iteration k. The process was stopped when $|e_k| \leq 10^{-4}$. The cubic local convergence of the algorithm based on S_1 is clearly observed in this example. Let us explain the reasons why the iteration function S_1 behaves much better than S_2 and S_3. The behavior of S_2 is poor because γ is not concave, so the maximizer of the approximating quadratic does not always exist, and sometimes we must use the safeguarded step. The referee pointed out that one should expect

TABLE 1
Numerical example.

Method k	S_1	S_2	S_3
0	− 2.33622072	−2.33622072	−2.33622072
1	0.21082107	12.1403186	242.321953
2	−8.15242529 E − 5	0.895310618	3.36624916
3		−1.41726372	0.358263649
4		−0.454572484	9.65546072 E − 3
5		0.168417811	−1.64439996
6		−0.15556109	−0.947321124
7		3.24829039 E − 3	−0.503044806
8		−0.769436508	−0.255501084
9		−0.370453373	6.51705265 E − 3
10		−0.169479772	3.6880374 E − 6
11		−6.86214119 E − 3	
12		−1.80991739 E −3	
13		7.18511641 E − 4	
14		−5.45911491 E − 4	
15		−8.62553716 E − 5	

a similar behavior of S_3 and the Hebden–Moré algorithm for the global case, since both use the same type of approximation of γ. However, we observe that, typically, γ behaves very similarly to a linear function for $\mu \in (-\lambda_1, \infty)$, while this is not the case in the region $[-\lambda_2, -\lambda_1]$, where, generally, γ has a maximizer and the linear approximation is very poor if the root is close to this maximizer. S_2 and S_3 produce iterates that are close to the solution at iterations 7 and 4, respectively, but after that, their performance deteriorates due to the necessity of safeguarded iterations.

5. Final remarks. As we mentioned in the Introduction, we think that the main utility of the characterization of local minimizers given in this paper is its application to the minimization of a general quadratic on a region defined by the intersection of two balls. The latter problem appears when one applies a trust-region strategy to the minimization of a general differentiable function with a ball constraint (see Martínez and Santos [1991]). Heinkenschloss [1990] has used this approach for parameter identification problems.

At the present state of research, the main limitation of this strategy is that the technique introduced in §4 requires good approximations of the first two eigenvalues of the matrix. So, the process of finding global and local minimizers is more expensive than generally required by subproblems of optimization algorithms. This means that we are constrained to small to medium scale problems, where the cost of evaluating the function is high compared with the cost of the subproblem. A similar limitation is encountered when one considers the implementation of tensorial methods for unconstrained optimization. (See Schnabel and Chow [1991].) Those characteristics appear in practical applications, including some parameter identification problems.

Acknowledgments. Most of this work was done while I was visiting the Mathematical Sciences Department of Rice University in 1989. I acknowledge professors John Dennis and Richard Tapia for many helpful discussions and for their hospitality during that period. I also acknowledge Sandra A. Santos, for her work in continuation of this research. Finally, I am indebted to the referee, who found an error in the main result of §3 and wrote a careful report that helped me to improve the final version of this paper.

REFERENCES

M. R. CELIS, J. E. DENNIS, AND R. A. TAPIA [1984], *A trust region strategy for nonlinear equality constrained optimization*, in Numerical Optimization, P. T. Boggs, R. Byrd, and R. Schnabel, eds., Society for Industrial and Applied Mathematics, Philadelphia, PA, pp. 71–82.

J. E. DENNIS, JR., D. M. GAY, AND R. E. WELSCH [1981], *An adaptive nonlinear least-squares algorithm*, ACM Trans Math. Software, 7, pp. 348–368.

J. E. DENNIS, H. J. MARTÍNEZ, AND R. A. TAPIA [1988], *A convergence theory for the structured BFGS secant method with an application to nonlinear least squares*, Tech. Rep. 87-15, Mathematical Sciences Dept., Rice University.

J. E. DENNIS, J. M. MARTÍNEZ, R. A. TAPIA, AND K. A. WILLIAMSON [1990], *An algorithm based on a convenient trust-region subproblem for nonlinear programming*, Tech. Rep., Mathematical Sciences Dept., Rice University.

J. R. ENGELS AND H. J. MARTÍNEZ [1991], *Local and superlinear convergence for partially known quasi-Newton methods*, SIAM J. Optimization, 1, pp. 42–56.

R. FLETCHER [1987], *Practical Methods of Optimization*, 2nd ed., John Wiley, Chichester, New York.

D. M. GAY [1981], *Computing optimal locally constrained steps*, SIAM J. Sci. Stat. Comput., 2, pp. 186–197.

S. M. GOLDFELD, R. E. QUANDT, AND H. F. TROTTER [1966], *Maximization by quadratic hill-climbing*, Econometrica, 34, pp. 541–551.

G. H. GOLUB AND C. F. VAN LOAN [1989], *Matrix Computations*, The Johns Hopkins University Press, Baltimore, London.

M. D. HEBDEN [1973], *An algorithm for minimization using exact second derivatives*, Tech. Rep. TP515, AERE Harwell, England.

M. HEINKENSCHLOSS [1990], *Mesh independence for nonlinear least squares problems with norm contraints*, Tech. Rep. 90-18, Dept. of Mathematical Sciences, Rice University.

D. G. LUENBERGER [1984], *Linear and Nonlinear Programming*, 2nd ed., Addison-Wesley, Reading, MA.

J. M MARTÍNEZ [1990], *Local Convergence Theory of inexact Newton methods based on structured least change updates*, Math. Comput., 55, pp. 143–168.

J. M. MARTÍNEZ AND S. A. SANTOS [1991], *Trust-region algorithms on arbitrary domains*, Contributed paper, TIMS-SOBRAPO Meeting, Rio de Janeiro, July.

J. J. MORÉ [1978], *The Levenberg-Marquardt algorithm: implementation and theory*, in Numerical Analysis, Dundee 1977, Lecture Notes in Mathematics 630, G. A. Watson, ed., Springer-Verlag, Berlin, pp. 105–116.

——— [1983], *Recent developments in algorithms and software for trust region methods*, in Mathematical Programming Bonn 1982. The State of the Art, A. Bachem, M Grötschel, and B. Korte, eds., Springer-Verlag, Berlin.

J. J. MORÉ AND D. C. SORENSEN [1983], *Computing a trust region step*, SIAM J. Sci. Statist. Comput., 4, pp. 553–572.

M. J. D. POWELL AND Y. YUAN [1991], *A trust region algorithm for equality constrained optimization*, Math. Programming, 49, pp. 189–211.

C. H. REINSCH [1971], *Smoothing by spline functions* II, Numer. Math., 16, pp. 451–454.

R. B. SCHNABEL AND T. CHOW [1991], *Tensor methods for unconstrained optimization using second derivatives*, SIAM J. Optimization, 1, pp. 293–315.

S. A. SANTOS [1991], *Métodos de Regiões de Confiança em domínios arbitrários e minimização em bolas*, Tese de Mestrado, Dep. de Matemática Aplicada, UNICAMP, Campinas, Brasil.

D. C. SORENSEN [1982], *Newton's method with a model trust region modification*, SIAM J. Numer. Anal., 19, pp. 409–426.

K. A. WILLIAMSON [1990], *A robust trust region algorithm for nonlinear programming*, Ph.D. thesis, Dept. of Mathematical Sciences, Rice University.

Y. YUAN [1988], *On a subproblem of trust region algorithms for constrained optimization*, Tech. Rep. NA 10, DAMPT, University of Cambridge, Cambridge, UK.

SIAM J. OPTIMIZATION
Vol. 4, No. 1, pp. 177–192, February 1994
© 1994 Society for Industrial and Applied Mathematics
010

ON THE MAXIMIZATION OF A CONCAVE QUADRATIC FUNCTION WITH BOX CONSTRAINTS*

ANA FRIEDLANDER† AND JOSÉ MARIO MARTÍNEZ†

Abstract. A new method for maximizing a concave quadratic function with bounds on the variables is introduced. The new algorithm combines conjugate gradients with gradient projection techniques, as the algorithm of Moré and Toraldo [*SIAM J. Optimization*, 1 (1991), pp. 93–113] and other well-known methods do. A new strategy for the decision of leaving the current face is introduced that makes it possible to obtain finite convergence even for a singular Hessian and in the presence of dual degeneracy. Numerical experiments are presented.

Key words. quadratic programming, conjugate gradient projection, active set methods, large scale

AMS subject classifications. 65K10, 90C20

1. Introduction.

In this paper, we consider the problem of maximizing a concave quadratic function subject to bounds on the variables. This problem (or its equivalent: minimizing a convex quadratic function on a box) appears frequently in applications, for instance in finite-difference discretization of free boundary problems (Cea and Glowinski [1983], O'Leary [1980]), numerical simulation of friction problems in rigid body mechanics (Lötstedt [1984]), image reconstruction from projections (Herman [1980]), implementation of robust methods for nonlinear programming (Nickel and Tolle [1989]), etc.

Many successful algorithms for solving this type of problem are based on active set strategies (see Gill, Murray, and Wright [1981] and Fletcher [1987]). Briefly, an active set method proceeds by generating iterates on a face of the feasible polytope until either a maximizer of the objective function on that face or a point on the boundary of the face is reached. In the first case the iterate is allowed to leave the current face and the algorithm continues working on a face of a higher dimension. Since the function values are strictly increasing, finite convergence is obtained. Many times, finite convergence results are based on the finite termination properties of the conjugate gradient method for quadratic functions (Hestenes and Stiefel [1952], Golub and Van Loan [1989]). This classical approach has two main disadvantages: one is that constraints are added one at a time to the working set, a fact that leads to an excessive number of iterations in large scale problems. The other disadvantage is that the exact maximizer on the current working face is required before dropping constraints. Of course, obtaining the exact maximizer may require many conjugate gradient iterations if the dimension of the face is large.

In order to avoid the disadvantages of the active set strategy, a different type of algorithm, based on the gradient projection, has been proposed by several authors. Bertsekas [1982] used a scaled gradient projection method for minimizing arbitrary functions subject to bound constraints; Dembo and Tulowitzki [1987] proposed the combination of the gradient projection method with conjugate gradients; and, more recently, Moré and Toraldo [1991] introduced an algorithm of the same type that has

* Received by the editors October 28, 1991; accepted for publication (in revised form) November 30, 1992. This work was supported by FAPESP grant 90/3724/6, FINEP, CNPq, and FAEP-UNICAMP.

† Department of Applied Mathematics, IMECC–UNICAMP, State University of Campinas, CP 6065, 13081 Campinas, SP, Brazil.

finite convergence if the problem is strictly convex and the solution is nondegenerate. Other authors, including Yang and Tolle [1986] and Wright [1989], followed the ideas of Dembo and Tulowitzki to introduce algorithms whose efficiency is currently under discussion.

In this paper, we present a new algorithm that combines an active set strategy with the gradient projection method. We avoid the necessity of finding an exact maximizer on a face by testing a computable criterion on the current iteration which, if satisfied, guarantees that the next point will have a higher function value than the maximum value of the function on a neighborhood of fixed size, restricted to the current face. This property is essential to prove finite convergence for a (not necessarily strictly) concave quadratic without nondegeneracy assumptions. The gradient projection techniques are used to speed up the process of identifying the optimal face, but they play no role in the convergence analysis. Inside each face, we use the conjugate gradient method.

We tested the new algorithm with the obstacle problem, for which we obtained results comparable to those reported by Dembo and Tulowitzki and Moré and Toraldo. Our results seem to be the best in problems with near degenerate solutions. In another collection of randomly generated tests our algorithm showed the same typical performance: few faces are visited before the optimal face is reached, and the behavior is insensitive to degeneracy. We observed the same behavior in a third collection of test problems related to image reconstruction, where the Hessian is always singular.

This paper is organized as follows: In §2 we describe the algorithm. In §3 we prove the convergence results. In §§4 and 5 we describe some important features of the computer implementation. In §6 we present the computational experiments. Some conclusions and guidelines for future research are given in §7.

2. Basic method. We consider the problem of maximizing a concave quadratic funtion with bound constrained variables:

$$\begin{aligned} &\text{Maximize } q(x) \\ &\text{s.t. } x \in \Omega, \end{aligned} \tag{2.1}$$

where $\Omega = \{x \in I\!R^n \mid \ell \leq x \leq u, \ell < u\}, q(x) = \frac{1}{2}x^T Hx + b^T x$, and H is negative semidefinite.

We denote $g(x) \equiv \nabla q(x) \equiv Hx + b$ for all $x \in I\!R^n$. Let $L > 0$ be such that $||H|| \leq L$. ($||\cdot||$ denotes the 2-norm of vectors or matrices.)

It is easy to see that

$$0 \geq q(z) - q(x) - \nabla q(x)^T(z-x) = \tfrac{1}{2}(z-x)^T H(z-x) \geq (-L/2)||z-x||^2 \tag{2.2}$$

for all $x, z \in I\!R^n$.

We define an open face of Ω as a set $F_I \subset \Omega$ such that

$$\begin{aligned} &I \text{ is a (possibly empty) subset of } \{1, 2, \ldots, 2n\} \text{ such that } i \text{ and } n+i \\ &\text{cannot belong simultaneously to } I \text{ for all } i \in \{1, 2, \ldots, n\} \end{aligned} \tag{2.3}$$

and

$$F_I = \{x \in \Omega \mid x_i = \ell_i \text{ if } i \in I,\ x_i = u_i \text{ if } n+i \in I,\ \ell_i < x_i < u_i \text{ otherwise}\}. \tag{2.4}$$

Therefore, the set Ω is divided into 3^n disjoint faces. Let us call $\overline{F}_I$ the closure of each open face, $[F_I]$ the smallest linear manifold which contains F_I, $S(F_I)$ the parallel

subspace to $[F_I]$, and $\dim F_I$ the dimension of $S(F_I)$. Clearly, $\dim F_I = n - \#I$. For each $x \in \Omega$ let us define the projected gradient $g_P(x) \in \mathbb{R}^n$ as

$$g_P(x)_i = 0 \quad \text{if} \quad x_i = \ell_i \quad \text{and} \quad \frac{\partial q}{\partial x_i}(x) < 0 \tag{2.5}$$

$$\text{or}$$

$$x_i = u_i \quad \text{and} \quad \frac{\partial q}{\partial x_i}(x) > 0,$$

$$= \frac{\partial q}{\partial x_i}(x) \quad \text{otherwise.}$$

A necessary and sufficient condition for x being a global solution of (2.1) is

$$g_P(x) = 0. \tag{2.6}$$

For each $x \in \overline{F}_I$ let us define $g_I(x) \in \mathbb{R}^n$ as

$$g_I(x)_i = 0 \quad \text{if} \quad i \in I \quad \text{or} \quad n+i \in I, \tag{2.7}$$

$$= \frac{\partial q}{\partial x_i}(x) \quad \text{otherwise.}$$

Thus, $g_I(x)$ is the orthogonal projection of $\nabla q(x)$ on $S(F_I)$. We also define, for $x \in \overline{F}_I$,

$$g_I^C(x)_i = 0 \quad \text{if} \quad i \notin I \quad \text{and} \quad n+i \notin I, \tag{2.8}$$

$$= 0 \quad \text{if} \quad i \in I \quad \text{and} \quad \frac{\partial q}{\partial x_i}(x) < 0$$

$$\text{or}$$

$$n+i \in I \quad \text{and} \quad \frac{\partial q}{\partial x_i}(x) > 0,$$

$$= \frac{\partial q}{\partial x_i}(x) \quad \text{otherwise.}$$

The vector $g_I^C(x)$ was introduced in Friedlander and Martínez [1989] and named "chopped gradient" related to F_I. It plays a major role in the main results of this paper. Clearly, for all $x \in F_I$, we have

$$g_P(x) = g_I(x) + g_I^C(x).$$

LEMMA 2.1. *Assume that* $\overline{x} \in \overline{F}_I$ *is such that*

$$q(\overline{x}) \geq q(x) \quad \textit{for all} \quad x \in \overline{F}_I. \tag{2.9}$$

Then the following statements are equivalent:

$$q(\overline{x}) \geq q(x) \quad \textit{for all} \quad x \in \Omega, \tag{2.10}$$

and

$$g_I^C(\overline{x}) = 0. \tag{2.11}$$

Proof. Let us assume (2.10). If $i \in I$, then $\overline{x}_i = \ell_i$, and by (2.5) and (2.6), $(\partial q/\partial x_i)(\overline{x}) \leq 0$. Similarly, if $n+i \in I$ then $\overline{x}_i = u_i$ and $(\partial q/\partial x_i)(\overline{x}) \geq 0$. Therefore, by (2.8), $g_I^C(\overline{x}) = 0$.

Now assume (2.11). We wish to prove that $g_P(\overline{x}) = 0$.

For $i \in \{1, 2, \ldots, n\}$, let us consider each of the three following possibilities:

$$\overline{x}_i = \ell_i, \tag{2.12}$$

$$\overline{x}_i = u_i, \tag{2.13}$$

and

$$\ell_i < \overline{x}_i < u_i. \tag{2.14}$$

If (2.12) is true, we have two alternatives:

$$i \in I \tag{2.15}$$

and

$$i \notin I. \tag{2.16}$$

If (2.15) holds, (2.8) and $g_I^C(\overline{x})_i = 0$ imply that $(\partial q/\partial x_i)(\overline{x})_i \leq 0$. Then, by (2.5), $g_P(\overline{x})_i = 0$. If (2.16) holds, $x_i > \ell_i$ for all $x \in F_I$. But, by (2.9), $q(\overline{x}) \geq q(x)$ for all $x \in \overline{F}_I$. Therefore, $(\partial q/\partial x_i)(\overline{x}) \leq 0$ and so, by (2.5), $g_P(\overline{x})_i = 0$.

The same argument leads to $g_P(\overline{x})_i = 0$ when $\overline{x}_i = u_i$. Now, if (2.14) holds, we have by (2.9) that $(\partial q/\partial x_i)(\overline{x}) = 0$. Thus, the desired result is proven. □

In Lemma 2.1 we proved that a stationary point $\overline{x}$ for $\overline{F}_I$ either is a global solution of (2.1), or has a nonnull $g_I^C(\overline{x})$. Thus $g_I^C(\overline{x})$ should be a useful direction for escaping from a nonoptimal face. Observe that x can belong to different "closed faces" $\overline{F_I}$, so different vectors $g_I^C(x)$ $(g_I(x))$ can be defined at a single point. However, only the vector $g_I^C(x)$ $(g_I(x))$ that corresponds to the unique *open* face F_I containing x will be used in the algorithms.

Now, we are able to define the following algorithm.

ALGORITHM 2.1. MAIN MODEL ALGORITHM. Let $L > 0$ be such that $||H|| \leq L$. Let $x^0 \in \Omega$ be a given initial point, $\delta > 0$.

If $x^k \in F_I$ $(I = I_k = I(x^k))$ is the kth approximation to a maximizer of q in Ω and $g_P(x^k) \neq 0$, we define $\gamma_I \equiv \min\{u_i - \ell_i \mid i \in I \text{ or } n + i \in I\}$ and we obtain x^{k+1} by the following steps.

Step 1. Test if $||g_I(x^k)||$ is small enough for leaving the face.

Choose $D_k \geq \delta$.

If $||g_I^C(x^k)|| > L\gamma_I$ and $||g_I(x^k)|| < L\gamma_I^2/(2D_k)$

or

$||g_I^C(x^k)|| \leq L\gamma_I$ and $||g_I(x^k)|| < ||g_I^C(x^k)||^2/(2D_kL)$,

go to Step 2. Else, go to Step 3.

Step 2. Find a new point not belonging to $\overline{F}_I$.

Compute $x^{k+1} \in \Omega - \overline{F}_I$ such that

$$q(x^{k+1}) > q(x^k) + ||g_I(x^k)||D_k. \tag{2.17}$$

Step 3. Compute the conjugate gradient direction.

If $k = 0$ or $x^{k-1} \notin F_I$, define $d_k = g_I(x^k)$.

Else, define

$$d_k = g_I(x^k) + \beta_k d_{k-1},$$

where

$$\beta_k = ||g_I(x^k)||^2/||g_I(x^{k-1})||^2.$$

Step 4. Find a new point on $\overline{F}_I$.

If $q(x)$ is unbounded above along the half-line $L_k \equiv \{x^k + \lambda d_k, \lambda \geq 0\}$ or if the maximizer y^k of $q(x)$ on L_k does not belong to F_I, then obtain x^{k+1} as any point on $\overline{F}_I - F_I$ that satisfies

$$q(x^{k+1}) > q(x^k). \tag{2.18}$$

Else, define $x^{k+1} = y^k$.

Remark. In §3 we will prove that Algorithm 2.1 is well defined and that it finds a solution x^* in a finite number of iterations. Now we intend to clarify the meaning of the steps that define the algorithm.

The main property of the algorithm is that when a closed face $\overline{F}_I$ is abandoned, no future iterate will belong to a ball with center x^k and radius D_k in $\overline{F}_I$. Since q is concave, we have that

$$q(x) \leq q(x^k) + g_I(x^k)^T(x - x^k) \leq q(x^k) + ||g_I(x^k)||D_k \tag{2.19}$$

for all $x \in \overline{F}_I$ such that $||x - x^k|| \leq D_k$, and so, the property above follows from the monotonicity of $q(x^k)$. At Step 1, we test sufficient conditions for obtaining the required increase at Step 2.

In fact, let us consider the quadratic function defined by

$$\phi(\lambda) = q(x^k) + \lambda||g_I^C(x^k)||^2 - \lambda^2(L/2)||g_I^C(x^k)||^2.$$

From (2.2) we have that, for $\lambda \geq 0$,

$$q(x^k + \lambda g_I^C(x^k)) \geq \phi(\lambda).$$

The unconstrained maximizer of ϕ is $\lambda = 1/L$. Moreover, the maximizer of ϕ for $\lambda \in [0, \gamma_I/||g_I^C(x^k)||]$ is $\widetilde{\lambda} = \min\{1/L, \gamma_I/||g_I^C(x^k)||\}$. This means that the function q may be increased by an amount of at least

$$\Delta = \widetilde{\lambda}||g_I^C(x^k)||^2 - \widetilde{\lambda}^2(L/2)||g_I^C(x^k)||^2$$

if the new point is the maximizer of q on the segment $[x^k, x^k + \gamma_I g_I^C(x^k)/||g_I^C(x^k)||]$. At Step 1 of Algorithm 2.1 we essentially test if $\Delta \geq ||g_I(x^k)||D_k$. If this happens, it turns out that the desired increase can be obtained. Procedures for obtaining this increase will be described in §4. However, we will see in the proof of Theorem 3.1 that, for getting the desired increase, it is sufficient that eventually the choice $x^k + \widetilde{\lambda}||g_I^C(x^k)||$ be made.

On the other hand, if $\Delta < ||g_I(x^k)||D_k$, we judge that the set $\overline{F}_I$ should be better explored, and our search for a new point is restricted to this set, with the single requirement that $q(x^{k+1}) > q(x^k)$. We will also describe in §4 the procedure for finding x^{k+1} when the maximizer along L_k does not exist or when it lies outside F_I.

The search direction when two consecutive points are in the same open face F_I and when the condition for leaving the face is not satisfied is a conjugate gradient direction internal to the face.

One of the referees pointed out that a conceptual difference between Algorithm 2.1 and the other algorithms that combine gradient projection and conjugate gradient techniques is that our algorithm does not try to find a good feasible point on the current face and move to another face when it is discovered that the face is not optimal. Instead, with the test of Step 1, we determine if a change of face will be beneficial and, if so, the set I is modified and a new face is explored. Our strategy has two consequences: one of them favors staying on the current face, and the other favors abandoning it. On one hand, we only abandon the current face if we can *guarantee* that a point outside exists that has a functional value greater than the maximum of q on a ball of radius D_k inside the face. Depending on the sharpness of the bound L, it is possible that the desired increase along g_P or g_I^C could be obtained at the current iteration, even with a failure of the test. On the other hand, when the test recommends leaving the face, it is possible that the desired increase could be obtained inside the face, evidently at a point $\overline{x}$ such that $||\overline{x} - x^k|| > D_k$. The probability of this occurrence decreases as D_k increases, and the increase can be obtained only by abandoning the face if D_k is larger than its diameter. We take the point of view that the comparisons of Step 1 represent a measure of the relation between the magnitudes $||g_P(x^k)||$ (or $||g_I^C(x^k)||$) and $||g_I(x^k)||$. If $||g_I(x^k)||$ is small compared to $||g_P(x^k)||$ (or $||g_I^C(x^k)||$) we must abandon the face, and vice versa. The extreme cases are evident: if $||g_I^C(x^k)|| = 0$ we stay on the face and if $||g_I(x^k)|| = 0$ we abandon it. Inequalities (2.17) and (2.19) give a geometrical meaning for intermediate situations. We will see that our strategy also has interesting theoretical consequences.

3. Convergence results. In this section we prove the following theorems.

THEOREM 3.1. *Algorithm* 2.1 *is well defined.*

THEOREM 3.2. *If* $\{x^k\}$ *is a sequence generated by Algorithm* 2.1, *then* $\{x^k\}$ *terminates at a solution* x^* *of problem* (2.1) *in a finite number of steps.*

Theorems 3.1 and 3.2 are obtained as a consequence of the following lemmas.

LEMMA 3.1. *Let* $\gamma_I = \min\{u_i - \ell_i \,|\, i \in I \text{ or } n+i \in I\}$ *and* $x \in \overline{F}_I$ *such that* $g_I^C(x) \neq 0$. *Define*

$$\omega_I^C(x) = g_I^C(x)/||g_I^C(x)||. \tag{3.1}$$

Then

$$x + \gamma\omega_I^C(x) \in \Omega - \overline{F}_I \quad \text{for all } \gamma \in (0, \gamma_I]. \tag{3.2}$$

Proof. We prove first that

$$\ell_i \leq x_i + \gamma\omega_I^C(x) \leq u_i \tag{3.3}$$

for all $i \in \{1, 2, \ldots, n\}$, $\gamma \in (0, \gamma_I]$.

If $i \notin I$ and $n + i \notin I$, (3.3) is true since $\omega_I^C(x)_i = 0$ by (2.8).

If $i \in I$, we have, since $x \in \overline{F}_I$, that $x_i = \ell_i$.

Therefore, by (2.8), $\omega_I^C(x)_i \geq 0$. By (3.1) we have

$$0 \leq \omega_I^C(x)_i \leq 1; \tag{3.4}$$

thus,

$$\ell_i \leq \ell_i + \gamma\omega_I^C(x)_i \leq \ell_i + \gamma_I \leq \ell_i + (u_i - \ell_i) \leq u_i.$$

If $\omega_I^C(x)_i > 0$, then $\ell_i < \ell_i + \gamma\omega_I^C(x)_i$, which implies that $x + \gamma\omega_I^C(x) \notin \overline{F}_I$. A similar argument is used if $n + i \in I$; therefore, $x + \gamma\omega_I^C(x) \in \Omega$. But since $g_I^C(x) \neq 0$, $\omega_I^C(x)$ necessarily has a nonzero component. So $x + \gamma\omega_I^C(x) \notin \overline{F}_I$. This completes the proof. □

LEMMA 3.2. *Define γ_I as in Lemma 3.1. Let $\lambda \in (0, \gamma_I]$ and $x \in \overline{F}_I$ be such that $g_I^C(x) \neq 0$. Then for all $y \in F_I$ such that $||x - y|| \leq D_k$,*

$$q(x + \lambda\omega_I^C(x)) - q(y) \geq \lambda||g_I^C(x)|| - (L/2)\lambda^2 - ||g_I(x)||D_k. \tag{3.5}$$

Proof. The concavity of q implies that, for all $y \in F_I$,

$$q(y) \leq q(x) + g_I(x)^T(y - x). \tag{3.6}$$

So, by the Cauchy–Schwarz inequality, we have, for all $y \in F_I$ such that $||x-y|| \leq D_k$,

$$q(y) \leq q(x) + ||g_I(x)||D_k. \tag{3.7}$$

Now, by (2.2), we have for all λ,

$$q(x + \lambda\omega_I^C(x)) - q(x) - \nabla q(x)^T\lambda\omega_I^C(x) \geq (-L/2)\lambda^2. \tag{3.8}$$

Therefore,

$$q(x + \lambda\omega_I^C(x)) - q(x) \geq \lambda||g_I^C(x)|| - (L/2)\lambda^2. \tag{3.9}$$

Substracting (3.7) from (3.9) we obtain (3.5). □

Proof of Theorem 3.1. If $x^k \in F_I$ is not a solution of problem (2.1), x^{k+1} is defined at either Step 2 or Step 4.

We get into Step 2 if

$$||g_I^C(x^k)|| > L\gamma_I \quad \text{and} \quad L\gamma_I^2/(2D_k) > ||g_I(x^k)|| \tag{3.10}$$

or

$$||g_I^C(x^k)|| \leq L\gamma_I \quad \text{and} \quad ||g_I^C(x^k)||^2/(2D_kL) > ||g_I(x^k)||. \tag{3.11}$$

By Lemma 3.1 we have that $x^k + \gamma_I\omega_I^C(x^k) \in \Omega - \overline{F}_I$.

If (3.10) holds, by (3.9), we have that

$$\begin{aligned} q(x^k + \gamma_I\omega_I^C(x^k)) &\geq q(x^k) + \gamma_I||g_I^C(x^k)|| - (L/2)\gamma_I^2 \\ &> q(x^k) + L\gamma_I^2 - (L/2)\gamma_I^2 \\ &> q(x^k) + ||g_I(x^k)||D_k. \end{aligned}$$

If (3.11) holds, define $\rho = ||g_I^C(x^k)||/L$. Then, by (3.9) and (3.11), we have that

$$\begin{aligned} q(x^k + \rho\omega_I^C(x^k)) &\geq q(x^k) + ||g_I^C(x^k)||^2/L - ||g_I^C(x^k)||^2/(2L) \\ &= q(x^k) + ||g_I^C(x^k)||^2/(2L) \\ &> q(x^k) + ||g_I(x^k)||D_k. \end{aligned}$$

Therefore, when x^{k+1} is computed at Step 2, we have that

$$\begin{aligned} &\max_{\lambda\in[0,\gamma_I]}\{q(x^k + \lambda\omega_I^C(x^k))\} \\ &\quad = \max_{\lambda\geq 0}\{q(x^k + \lambda g_I^C(x^k)) \mid x^k + \lambda g_I^C(x^k) \in \Omega\} \\ &\quad > q(x^k) + ||g_I(x^k)||D_k. \end{aligned} \tag{3.12}$$

The objective of Step 2 is to obtain a point that satisfies (2.17). By (3.12) we see that such a point exists. In fact, we proved that the maximizer of $q(x)$ on the feasible part of the half-line defined by $g_I^C(x^k)$ satisfies that inequality. So, any algorithm that eventually computes q at that trial point can be used at Step 2.

Finally, a point x^{k+1} satisfying (2.18) at Step 4 exists because the d_k's are ascent directions. □

LEMMA 3.3. *If x^{k+1} is defined at Step 2 of Algorithm 2.1 and $x \in \overline{F}_I$ is such that $||x - x^k|| \leq D_k$, then $q(x^{k+1}) > q(x)$.*

Proof. This result follows from (2.17) and (2.19). □

LEMMA 3.4. *Assume that $\overline{F}_I$ does not contain a global optimizer of problem* (2.1) *and that $x^k \in F_I$; then there exists $j \geq 1$, such that $x^{k+j} \notin F_I$.*

Proof. Suppose, on the contrary, that $x^{k+j} \in F_I$ for all $j \geq 1$. Let $k_0 \leq k$ be the smallest index such that $x^\ell \in F_I$ for all $\ell \geq k_0$. Therefore, $d_{k_0} = g_I(x^{k_0})$. Moreover, the successive iterations $x^{k_0+1}, x^{k_0+2}, \ldots$, are conjugate gradient iterations inside F_I. So, by Hestenes and Stiefel [1952] and Yang and Tolle [1986], there exists $m > 0$ such that $x^{k_0+m} \in F_I$ and $g_I(x^{k_0+m}) = 0$. But, in this case, x^{k_0+m} satisfies the conditions of Step 1 of Algorithm 2.1 for leaving the face. Therefore, $x^{k_0+m+1} \notin F_I$, which is a contradiction. □

LEMMA 3.5. *For all $k \in \mathbb{N}$ such that x^k is not a global optimizer of* (2.1), *the following assertions hold:*

(A1) *x^{k+1} is well defined and $q(x^{k+1}) > q(x^k)$.*

(A2) *If $x^k \in F_I$, one and only one of the three following possibilities is true:*

(3.13) (i) $x^{k+1} \in F_I$;

(3.14) (ii) *There exists $J \subset \{1, \ldots, 2n\}$ such that $x^{k+1} \in F_J \subset \overline{F}_I - F_I$;*

(3.15) (iii) $x^{k+1} \notin \overline{F}_I$ *and* $q(x^{k+1}) > q(x)$ *for all* $x \in \overline{F}_I$ *such that* $||x - x^k|| \leq D_k$.

(A3) *If $x^k \in F_I$ and $\overline{F}_I$ does not contain a global maximizer of* (2.1), *there exists $\ell > k$ such that $x^\ell \notin F_I$.*

Proof. (A1), (A2), and (A3) are direct consequences of the previous lemmas and the definition of Algorithm 2.1. □

LEMMA 3.6. *There exists $\ell_0 \in \mathbb{N}$ such that, for all $k \geq \ell_0$ such that x^k is not a global solution of* (2.1), *either* (3.13) *or* (3.14) *hold.*

Proof. Suppose, on the contrary, that there exists an infinite set of indices $K_1 \subset \mathbb{N}$ such that for all $k \in K_1$, x^k is not a global maximizer of (2.1) and x^{k+1} satisfies (3.15). Since the number of different faces F_I is finite, there exists an infinite set $K_2 \subset K_1$ such that x^k belongs to a fixed F_I for all $k \in K_2$. So, for all $k \in K_2$, we have that $q(x^{k+1}) > q(x)$ for all $x \in F_I$ such that $||x - x^k|| \leq D_k$. But $D_k \geq \delta > 0$. Therefore, $q(x^{k+1}) > q(x)$ for all $k \in K_2$, $x \in F_I$, $||x - x^k|| \leq \delta$. Hence, for any two different $k_1, k_2 \in K_2$, $k_1 < k_2$, we necessarily have that $||x^{k_1} - x^{k_2}|| \geq \delta > 0$. This is impossible, since F_I is bounded. Therefore, the desired result is proved. □

Proof of Theorem 3.2. By Lemma 3.6 we know that there exist ℓ_0 and F_I such that $x^k \in F_I$ for all $k \geq \ell_0$. Thus, by (A3), $\overline{F}_I$ contains a global maximizer of (2.1). Let k_0 be the first index such that $x^k \in F_I$ for all $k \geq k_0$. Then $d_{k_0} = g_I(x^{k_0})$ and all the subsequent points are obtained by conjugate gradient iterations inside F_I. Thus, by the classical results of Hestenes and Stiefel [1952] and the theorem of Yang and Tolle [1986] for the case where the Hessian is singular, we have that for some

$k \geq k_0$, $g_I(x^k) = 0$. But, if $g_I^C(x^k) \neq 0$, x^{k+1} would be defined and $x^{k+1} \notin \overline{F}_I$. This is impossible, so $g_I^C(x^k) = 0$. Then, by Lemma 2.1, x^k is a global maximizer of (2.1). □

4. Search on the piecewise linear path. In this section we specify how we effectively implement Steps 2 and 4 of the Model Algorithm 2.1. Remember that at Step 2, we require that the new point satisfy (2.17), while at Step 4, we only require (2.18). However, both steps will be implemented using a backtracking search along the polygonal path defined by a search direction. There are only minor differences between our projected search and the one of Moré and Toraldo [1991], due to the fact that a singular Hessian may occur in our case, causing unboundedness of q along the corresponding line, and to the fact that we do not use an Armijo condition for stopping the search, as Moré and Toraldo do.

We define the projection $P : \mathbb{R}^n \to \Omega$ by

$$P(x)_i = \begin{cases} x_i & \text{if } \ell_i \leq x_i \leq u_i, \\ u_i & \text{if } x_i > u_i, \\ \ell_i & \text{if } x_i < \ell_i. \end{cases}$$

Assume that x^k is the current kth approximation of the solution of the problem and that d_k is a feasible ascent direction. Given the search direction d_k, the new point $x^{k+1} \equiv P(x^k + \lambda_k d_k)$ is computed by a backtracking search along the polygonal path $\{P(x^k + \lambda d_k) \mid \lambda > 0\}$. If we are at Step 2 of Algorithm 2.1, we stop the search when $q(P(x^k + \lambda d_k)) > q(x^k) + ||g_I(x^k)||D_k$ and, if we are at Step 4 of Algorithm 2.1, the stopping criterion of the search is $q(P(x^k + \lambda d_k)) > q(x^k)$.

At Step 2 of Algorithm 2.1, if we define $d_k = g_I^C(x^k)$, we have already seen, in §3, that the maximizer of q in the first segment determined by d_k satisfies (2.17). Therefore, since in the backtracking search we eventually evaluate q at this point, it turns out that, with this definition of d_k, we obtain in finite time a point $x^{k+1} = P(x^k + \overline{\lambda} d_k)$ that satisfies (2.17). However, numerical experimentation and tradition recommend that we consider the projected gradient as a possible direction for defining the piecewise linear path. Frequently, the utilization of the direction $g_P(x^k)$ produces a point that satisfies (2.17). But, especially in nearly degenerate situations, this is not the case. Therefore, we cannot base our strategy for leaving the face only on the projected gradient, and so, the direction $g_I^C(x^k)$ should be eventually used. Summing up, let us define two possible strategies for leaving the face at Step 2 of Algorithm 2.1. The first one is based just on $g_I^C(x^k)$, and the second one tries $g_I^C(x^k)$ only in the case of a failure with $g_P(x^k)$. These two strategies are described in Algorithms 4.1 and 4.2.

ALGORITHM 4.1. STRATEGY FOR LEAVING THE FACE BASED ON $g_I^C(x^k)$.

Define $d_k = g_I^C(x^k)$. Perform the backtracking search and define $x^{k+1} = P(x^k + \overline{\lambda} d_k)$.

ALGORITHM 4.2. STRATEGY FOR LEAVING THE FACE BASED ON $g_P(x^k)$ AND $g_I^C(x^k)$.

Define $d_k = g_P(x^k)$. Perform the backtracking search.

If $q(P(x^k + \overline{\lambda} d_k)) > q(x^k) + ||g_I(x^k)||D_k$, then define $x^{k+1} = P(x^k + \overline{\lambda} d_k)$.

Else, execute Algorithm 4.1.

5. Strategy for the choice of D_k. The parameter D_k, which must be chosen at Step 1 of Algorithm 2.1, plays a major role in the overall behavior of the method. By the theory developed in §3, we know that when we escape from the current face at Step 2, then, by (2.17), the ball with center x^k and radius D_k will never be visited again. Therefore, if D_k is greater than the diameter of the face, we guarantee that the whole face will not be visited again. If D_k is small, the probability of returning to the current face is much greater. On the other hand, when D_k is large, the test for leaving the face is harder to satisfy, and a lot of conjugate gradient iterations inside the face could be necessary. If D_k is small, the test for leaving the face tends to be satisfied if $g_I^C(x^k) \neq 0$ and so, in general, the face is abandoned if the projected gradient is different from the internal gradient $g_I(x^k)$. So, if we choose a very small D_k for all $k = 0, 1, 2, \ldots$ and we use Algorithm 4.2 for leaving the face, the behavior of Algorithm 2.1 will be very similar (at least in nondegenerate situations) to a pure gradient projection method with conjugate gradient iterations for accelerating the process inside the open faces.

The freedom in the choice of D_k, allowed by Algorithm 2.1, can be exploited by many heuristic strategies that were proven useful in related methods. In fact, any algorithm that combines projected searches for leaving the face with conjugate gradient iterations necessarily faces the practical problem of giving efficient rules for deciding when to stay in the current face or when to leave it.

Our heuristic is an adaptation of a strategy introduced by Moré and Toraldo [1991]. The philosophy underlying this heuristic is that when the conjugate gradient iteration is making good progress, we should continue doing this type of iteration and, conversely, if the behavior of the conjugate gradient method is poor, the criterion for leaving the face should be relaxed with the hope that better function values could be obtained outside $\overline{F}_I$. Let D_I denote the diameter of F_I. At each iteration k, D_k will be defined as the minimum between D_I/M_k and α_k, where both M_k and α_k are updated at each iteration, according to the progress of the conjugate gradient method. Initial values $M_0 \geq 1$ and $\alpha_0 > 0$ must be given by the user, M_k is not allowed to increase over a large value $M(= 10^{10}$ in our experiments), and α_k cannot be less than a tolerance τ. If $r \geq 1, x^k, x^{k-1}, \ldots, x^{k-r} \in F_I$ and $q(x^k) - q(x^{k-1}) \geq 0.1 \max\{q(x^{k-\ell}) - q(x^{k-\ell-1}),\ \ell = 0, 1, \ldots, r-1\}$, we maintain $M_k = M_{k-1}$, $\alpha_k = \alpha_{k-1}$. Else, we set $M_k = \min\{M, 10M_{k-1}\}$, $\alpha_k = \max\{\tau, \alpha_{k-1}/10\}$, and $D_k = \min\{D_I/M_k, \alpha_k\}$. Therefore, D_k is always greater than the minimum between D_I/M and τ, so that there exists $\delta > 0$ such that $D_k \geq \delta$ for all k, as required by the Model Algorithm 2.1.

6. Numerical experiments. In this section, we study the practical behavior of our algorithm. We implemented our algorithm in double precision FORTRAN.

We use Algorithm 2.1, with the interpolatory backtracking projected searches described in §4, Algorithm 4.2 as strategy for leaving the face, and the choice of D_k described in §5. Alternatively, in some experiments, Algorithm 4.1 will be used instead of Algorithm 4.2. In some problems, the matrix H is explicitly available. In these cases, we choose $L = ||H||_\infty$. In other cases, H is not computed explicitly, but it has the form $H = A^T A$, where A is an $m \times n$ matrix. We use $L = ||A||_1 ||A||_\infty$ in these cases. In any practical situation, we recommend that L should be chosen as the best possible bound for $||H||$ that can be obtained with a low cost.

When, in a practical problem, a variable is unbounded (above or below) we impose an artificial (upper or lower) bound so that our algorithm can be applied. When both lower and upper bounds are natural, we recommend using a criterion based only on M_k to determine D_k at each iteration. In other words, in that case we use a very

large value of τ, so that $D_k = D_I/M_k$ for all k. On the other hand, if some of the bounds that determine the diameter of F_I are artificial, it is better to bound the size of the ball that will not be revisited by meaningful values of α_k since, in this case, D_I/M_k could be artificially large. Since all our experiments have natural bounds, we chose $\alpha_0 = \tau = 10^{10}$, so that M_0 is the only parameter that essentially determines the ball size D_k.

In all our tests the number of function evaluations was practically the same as the number of iterations. So, the computer time of the method in different computer environments was roughly proportional to the number of iterations. We used a PC-XT microcomputer with floating point co-processor and the Microsoft Fortran 77 compiler for running the first two sets of test problems. In these problems, the results for $n = 10000$ were obtained using a VAX 785 with the VMS operational system. In the third set of test problems we used a SUN Workstation.

Test Problem 1. The obstacle problem. (See Ciarlet [1978, pp. 287–296], Dembo and Tulowitzki [1987], and Moré and Toraldo [1991].) This problem consists of finding the equilibrium position of an elastic membrane which passes through a curve Γ. In our study, we considered three problems described by Dembo and Tulowitzki, which correspond to different bounds on the solution. We denote by $u(x, y)$ and $v(x, y)$ the lower obstacle function and the upper obstacle function, respectively. In problem A we set $u(x, y) = p_1[\sin(3.2x)\sin(3.3y)]^{p_2}$ and $v(x, y) = 2000$. In problem B, $u(x, y) = [\sin(9.2x)\sin(9.3y)]^{p_2}$ and $v(x, y) = u(x, y) + 0.02$. Finally, in problem C, $u(x, y) = [16x(1-x)y(1-y)]^{p_1}$ and $v(x, y) = u(x, y) + 0.01$.

We used the five-point finite-difference approximation to the Laplace operator. After discretization, the obstacle problem becomes a quadratic programming problem with box constraints, where the Hessian matrix H is such that $||H|| \leq 8$. So, we set $L = 8$ in the implementation of Algorithm 2.1.

We used the stopping criterion $||g_P(x^k)|| \leq 10^{-5}$. Higher precisions can be obtained, however, this is the stopping criterion used in Dembo and Tulowitzki [1987] and Moré and Toraldo [1991].

We ran many tests varying the values of M_0. The best results were obtained with a large value for M_0, which favors projected gradient iterations over conjugate gradient iterations. We will see that in other problems this is not the case.

We report the results of the experiments, for $M_0 = 10^9$.

We used the following starting points:

(a) $x^0 = \ell = (\ell_1, \ldots, \ell_n)^T$.

(b) $x^0 = \underline{1} = (1, \ldots, 1)^T$.

(c) $x^0 = u = (u_1, \ldots, u_n)^T$.

(d) $x^0 = (\ell + u)/2$.

The results of the experiments are given in Table 1. In this table, NI represents the number of iterations used to satisfy the convergence criterion and DT is the number of iterations reported by Dembo and Tulowitzki for their algorithm CGP. We observe that, in general, our results are better than those reported by Dembo and Tulowitzki . In problem A, with $p_1 = 0.3$ and $p_2 = 1$, the performance of the algorithm of Dembo and Tulowitzki is poor. They suggest that the reason for this poor performance is numerical dual degeneracy of the problem. Our algorithm behaved well in this case, perhaps because nondegeneracy is irrelevant for the convergence properties of the algorithm. Moré and Toraldo report the performance of their algorithm for problem A, with $p_1 = 1$, $p_2 = 1$, and $n = 10000$, and for other obstacle problems with dimensions different from those reported here. For this problem, it seems

TABLE 1
Experiments with the obstacle problem.

Problem	p_1	p_2	n	x^0	NI	DT
A	1	1	2601	ℓ	74	79
A	0.3	1	2601	ℓ	79	142
A	1	2	2601	ℓ	108	128
A	1	3	2601	ℓ	121	115
A	1	1	2601	$\underline{1}$	251	253
A	0.3	1	2601	$\underline{1}$	199	508
A	1	2	2601	$\underline{1}$	201	202
A	1	3	2601	$\underline{1}$	188	201
A	1	1	5041	ℓ	105	147
A	0.3	1	5041	ℓ	123	204
A	1	2	5041	ℓ	170	237
A	1	3	5041	ℓ	182	193
A	1	1	5041	$\underline{1}$	193	243
A	0.3	1	5041	$\underline{1}$	347	701
A	1	2	5041	$\underline{1}$	270	366
A	1	3	5041	$\underline{1}$	319	418
A	1	1	10000	ℓ	166	310
A	0.3	1	10000	ℓ	187	352
A	1	2	10000	ℓ	301	470
A	1	3	10000	ℓ	339	464
A	1	1	10000	$\underline{1}$	291	681
A	0.3	1	10000	$\underline{1}$	787	1733
A	1	2	10000	$\underline{1}$	337	596
A	1	3	10000	$\underline{1}$	541	661
B	3	2	5041	u	226	216
B	3	2	5041	ℓ	213	198
B	3	2	5041	$(u+\ell)/2$	139	143
C	3	2	5041	u	259	207
C	3	2	5041	ℓ	132	144
C	3	2	5041	$(u+\ell)/2$	116	122

that the overall performance of their algorithm is similar to ours. The spirit of the Moré–Toraldo method is different from ours in that they perform conjugate gradient iterations not only inside a face, but also out of its bounds, in order to find a suitable "truncated Newton" direction, internal to the current face. It could be argued that in many cases this strategy could waste a lot of time with infeasible points, but, in fact, it is not difficult to construct examples where finding the "truncated Newton" direction is more efficient than our strategy. Examples where the opposite behavior occurs are easy to obtain, as well. In any case, our criteria for leaving the face, which is the central contribution of this paper, can be easily adapted to truncated Newton directions of Moré–Toraldo type.

Test Problem 2. Random problems. We tested our algorithm with a set of random problems described in Moré and Toraldo [1989], with $n = 1000$. We call *nsing* the number of zero eigenvalues of H. In the nonsingular cases, the condition number of H is 10^3. Degenerate problems depend on a parameter *kdeg*, which indicates the number of zeros in $\nabla q(x^*)$ corresponding to active components of x^*. Problems with $kdeg > 0$ are degenerate. If $kdeg = 0$, the parameter *ndeg* is a measure of "near-degeneracy." The greater the value of *ndeg* the more "near degenerate" the problem is. $na(x^*)$ and $na(x^0)$ represent the number of active bounds at x^* and x^0, respectively.

The stopping criteria in all tests is $||g_P(x^k)|| \leq 10^{-10}$, and we used the obvious bound $L = ||H||$. As in the previous cases, we used $\alpha_0 = \tau = 10^{10}$, and we tested different values for M_0. Convergence was achieved in all the tests. In about 10% of the

tests, searches along g_I^C (calls to Algorithm 4.1 from Algorithm 4.2) were necessary in order to satisfy the criterion for leaving the face.

In all the nonsingular problems, we obtained $||x^* - x^k|| \leq 10^{-8}$, where x^k is the final iteration.

On singular problems the final x^k obtained by our method was always different from x^*. However, we obtained $|q(x^*) - q(x^k)| < 10^{-8}$ in all cases.

We observed that for nonsingular problems and $M_0 \in \{1, 100, 1000\}$, the number of iterations shows some sensitivity to degeneracy. This is not so clear for the number of faces visited, meaning that degeneracy may cause an increase of the work inside each face. In singular problems no essential difference with the nonsingular case appeared except that NI seems to be less sensitive to degeneracy. For degenerate problems the performance was very similar to that of the case $ndeg = 12$.

For $M_0 = 10^9$, a phenomenon similar to zig-zagging occurred in problems 16 and 18. In these two problems, $ndeg = 12$, and $na(x^*) = 900$, so that they have a high degree of "near-degeneracy" and a great number of constraints have to be added to the active set. In order to investigate the potentiality of the direction g_I^C of correcting zig-zagging, we solved problems 16 and 18 with $M_0 = 10^9$, but always using Algorithm 4.1 (g_I^C as first direction tried at Step 2 of Algorithm 2.1) instead of Algorithm 4.2. The results were very impressive. In problem 16, the number of iterations decreased from 1345 to 810 and the number of faces visited decreased from 489 to 149. In problem 18, convergence was achieved in 1096 iterations after visiting 191 faces when we used Algorithm 4.1, while, using Algorithm 4.2, 2091 iterations and 870 faces were necessary.

For $M_0 = 10^6$ we obtain the best results in terms of number of iterations (or computer time). The results are presented in Table 2. In the column NI we report the number of iterations performed and in the column NF we indicate the number of times a constraint is abandoned. We observed that, in many tests, NF was much lower for $M_0 = 10^3$ than for $M_0 = 10^6$. However, the number of iterations was lower for $M_0 = 10^6$, showing that NF is not a good parameter for measuring the efficiency of our algorithm. For $M_0 = 10^9$, NI was much greater than for $M_0 = 10^6$.

Test Problem 3. Function reconstruction from line integrals.

A set of box constrained quadratic problems was generated as follows. Assume that the interval $[0, 1]$ is divided into $ndiv$ intervals of equal size, $h = 1/ndiv$, and, consequently, the square $[0, 1] \times [0, 1]$ is divided into $n \equiv ndiv^2$ pixels. The problem is to find a function $u : [0, 1] \times [0, 1] \to [\ell, u]$, which is constant on each pixel, such that the line integrals along certain rays are known. See Herman [1980].

We generate the following sets of rays:

(a) Horizontal rays: $ndiv$ rays, from $(0, (i-1)h + h/2)$ to $(1, (i-1)h + h/2), i = 1, \ldots, ndiv$.

(b) Vertical rays: $ndiv$ rays, from $((i-1)h + h/2, 0)$ to $((i-1)h + h/2, 1), i = 1, \ldots, ndiv$.

(c) Rays forming an angle of 135 degrees with the x axis:

(i) $ndiv$ rays, from $(ih, 0)$ to $(0, ih), i = 1, \ldots, ndiv$.

(ii) $ndiv - 1$ rays, from $(1, ih)$ to $(ih, 1), i = 1, \ldots, ndiv - 1$.

(d) Rays forming an angle of 45 degrees with the x axis:

(i) $ndiv$ rays, from $((i-1)h, 0)$ to $(1, 1 - (i-1)h), i = 1, \ldots, ndiv$.

(ii) $ndiv - 1$ rays, from $(0, ih)$ to $(1 - ih, 1), i = 1, \ldots, ndiv - 1$.

Each ray defines one equation where only the variables corresponding to the pixels that are crossed by the ray are involved. Therefore, we have a sparse linear $m \times n$ system

TABLE 2
Experiments with random problems.

Problem	*nsing*	*kdeg*	*ndeg*	$na(x^*)$	$na(x^0)$	NI	NF
1	0	0	1	100	100	502	45
2	0	0	12	100	100	509	16
3	0	0	1	100	500	481	12
4	0	0	12	100	500	561	28
5	0	0	1	100	900	476	13
6	0	0	12	100	900	521	17
7	0	0	1	500	100	449	10
8	0	0	12	500	100	537	17
9	0	0	1	500	500	502	21
10	0	0	12	500	500	553	9
11	0	0	1	500	900	569	18
12	0	0	12	500	900	656	25
13	0	0	1	900	100	422	2
14	0	0	12	900	100	540	12
15	0	0	1	900	500	314	10
16	0	0	12	900	500	560	18
17	0	0	1	900	900	377	14
18	0	0	12	900	900	754	8
19	245	0	0	900	100	689	73
20	245	453	0	900	100	745	48
21	245	721	0	900	100	607	29
22	745	0	0	900	100	453	82

$Ax = d$, where $m = 6ndiv - 2$ and $n = ndiv^2$ with the constraints $\ell \leq x_i \leq u$ for all $i = 1, \ldots, n$. The unknowns x_i correspond to the values of the unknown function u on each pixel, numbered lexicographically. Finally, the problem is transformed in the box constrained quadratic optimization problem

$$\text{Maximize} - \tfrac{1}{2}||Ax - d||^2.$$

Obviously, the Hessian matrix $H = -A^TA$ is singular. It is easy to see that $||A||_\infty = 2^{1/2}$ and $||A||_1 = 4h2^{1/2}$. Therefore, we may use $L = 8h$ for running Algorithm 2.1. Observe that in all the algorithms described, only matrix-vector products Hv are required and so, the matrix H should not be stored explicitly.

In our experiments, we used $\ell = 0$ and $u = 1$. For generating the data (vector d) we define a function

$$u_* : [0,1] \times [0,1] \to [0,1]$$

and a function $\overline{u}$ which is constant on each pixel and coincides with u_* at the center of the pixel. The data vector d is formed by the line integrals of $\overline{u}$ along the rays. So, different experiments are characterized by different choices of u_*. We used the following functions.

Problem 1.

$$u_*(x,y) \equiv u_1(x,y) = 1 \quad \text{if } 0.25 \leq x \leq 0.75 \quad \text{and} \quad 0.25 \leq y \leq 0.75; \quad 0 \text{ otherwise.}$$

Problem 2.

$$u_*(x,y) \equiv u_2(x,y) = (x^2 + y)/2.$$

TABLE 3
Experiments with function reconstruction problems.

Problem	M_0	NI	NF
1	1	35	1
	100	35	1
	10^3	35	1
	10^6	35	18
	10^9	43	41
2	1	19	1
	100	19	1
	10^3	19	1
	10^6	19	1
	10^9	17	3
3	1	147	1
	100	147	1
	10^3	147	1
	10^6	253	156
	10^9	224	221

Problem 3.

$$u_*(x,y) = \min\{1, u_1(x,y) + u_2(x,y)\}.$$

In all the experiments, we used $x_i^0 = 0$ for all $i = 1, \ldots, n$, and $ndiv = 256$. So, the number of variables n is 65536.

The numerical results are given in Table 3. For each problem, we test different choices of M_0 and we report the number of faces visited for obtaining a solution. The stopping criterion was $||g_P(x^k)|| \leq 10^{-4}$.

7. Conclusions. The algorithm introduced in this paper seems to be a useful technique for solving quite general very large scale quadratic programming problems subject to bound constraints. Strict concavity is not necessary, and dual nondegeneracy is not required for finite convergence. Matrix factorizations are not used. These features are appealing for very large scale problems.

The convergence properties of our algorithm in the presence of dual degeneracy were confirmed by numerical experiments. This seems to be a very attractive feature since many successful algorithms for solving these problems have serious difficulties with near-degeneracy (see Coleman and Hulbert [1990]).

For theoretical reasons, our algorithm should be recommended when we suspect dual degeneracy or singularity of the Hessian. Clearly, we cannot predict dual degeneracy at the solution of a problem in advance, but singularity can be predicted in many practical situations, for example, when we have a linear $m \times n$ least squares problem with $m < n$, as in the case of our third set of experiments. Problems that come from the discretization of ill-posed inverse problems generally have singular or nearly singular Hessians. We are currently applying our algorithm to the regularization of this type of problem, with encouraging preliminary results. We think that for these problems, the strategy of using only feasible points is more efficient than the strategy of truncated Newton iterations inside the face, mentioned in §6. As a limit case, if the quadratic is almost linear, truncated Newton iterations would have no advantage at all.

As a general purpose algorithm for large-scale bound constrained quadratic programming, we recommend setting the important parameter M_0 greater than 10^3. In

fact, $M_0 = 10^3$ is our personal preference, but we were surprised by many tests where the best choice was much greater.

This work can be extended in two main directions (see Friedlander and Martínez [1991]). First, for nonquadratic concave problems we can prove that the convergence properties of natural generalizations (except finite termination) still hold. Second, the possibility of handling indefinite problems is a matter of current research.

Acknowledgments. We are indebted to two anonymous referees for their careful reading and interesting comments, which helped us to improve the quality of the paper.

REFERENCES

D. P. BERTSEKAS [1982], *Projected Newton methods for optimization problems with simple constraints*, SIAM J. Control Optim., 20, pp. 221–246.

J. CEA AND R. GLOWINSKI [1983], *Sur des Méthodes d'optimisation par relaxation*, RAIRO, R-3, pp. 5–32.

P. G. CIARLET [1978], *The Finite Element Method for Elliptic Problems*, North Holland, Amsterdam.

T. F. COLEMAN AND L. A. HULBERT [1989], *A direct active set algorithm for large sparse quadratic programs with simple bounds*, Math. Programming, 45, pp. 373–406.

——— [1990], *A globally and superlinearly convergent algorithm for convex quadratic programs with simple bounds*, Tech. Rep., 90-1092, Dept. of Computer Science, Cornell Univ., Ithaca, NY.

R. S. DEMBO AND U. TULOWITZKI [1987], *On the minimization of quadratic functions subject to box constraints*, Working paper series B 71, School of Organization and Management, Yale Univ.

R. FLETCHER [1987], *Practical Methods of Optimization*, 2nd ed., John Wiley, Chichester, New York.

A. FRIEDLANDER AND J. M. MARTÍNEZ [1989], *On the numerical solution of bound constrained optimization problems*, RAIRO Oper. Res., 23, pp. 319–341.

——— [1991], *New algorithms for maximization of concave functions with box constraints*, Relatório Técnico 48/89, IMECC-UNICAMP, Campinas, Brasil.

P. E. GILL, W. MURRAY, AND M. H. WRIGHT [1981], *Practical Optimization*, Academic Press, London, New York.

G. H. GOLUB AND C. F. VAN LOAN [1989], *Matrix Computations*, The Johns Hopkins University Press, Baltimore, London.

G. T. HERMAN [1980], *Image Reconstruction from Projections: The Fundamentals of Computerized Tomography*, Academic Press, New York.

M. R. HESTENES AND E. STIEFEL [1952], *Methods of conjugate gradients for solving linear systems*, J. Res. Nat. Bur. Standards, B49, pp. 409–436.

P. LÖTSTEDT [1984], *Solving the minimal least squares problem subject to bounds on the variables*, BIT, 24, pp. 206–224.

J. J. MORÉ AND G. TORALDO [1989], *Algorithms for bound constrained quadratic programming problems*, Numer. Math., 55, pp. 377–400.

——— [1991], *On the solution of large quadratic programming problems with bound constraints*, SIAM J. Optimization, 1, pp. 93–113.

R. H. NICKEL AND J. W. TOLLE [1989], *A sparse sequential programming algorithm*, J. Optim. Theory Appl., 60, pp. 453–473.

D. P. O'LEARY [1980], *A generalized conjugate gradient algorithm for solving a class of quadratic programming problems*, Linear Algebra Appl., 34, pp. 371–399.

S. J. WRIGHT [1989], *Implementing Proximal Point Methods for Linear Programming*, preprint MCS-P45-0189, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL.

E. K. YANG AND J. W. TOLLE [1986], *A class of methods for solving large, convex quadratic programs subject to box constraints*, Tech. Rep. UNC/ORSA/TR-86-3, Dept. of Operations Research and Systems Analysis, Univ. of North Carolina, Chapel Hill, NC.

SIAM J. OPTIMIZATION
Vol. 4, No. 1, pp. 193–207, February 1994
© 1994 Society for Industrial and Applied Mathematics
011

SHORT STEPS WITH KARMARKAR'S PROJECTIVE ALGORITHM FOR LINEAR PROGRAMMING*

J.-L. GOFFIN† AND J.-PH. VIAL‡

Abstract. A path-following, or short-step, version of Karmarkar's algorithm is proposed. When the first term of Karmarkar's potential function is given the weight $(n + n^{\delta})$, with $\delta \geq 0$, the algorithm converges in $O(n^{\max\{\delta/2, 1-\delta/2\}}L)$ iterations. The best convergence result $O(\sqrt{n}L)$ is achieved for $\delta = 1$. Two proofs of convergence are given: one based on duality gap reductions and the other on potential reductions. The weighted version of the standard projective algorithm is also analyzed. It is interpreted as a max-step algorithm. It is known to converge in at most $O((n+n^{\delta})L)$ iterations. The reduction of the duality gap when an updating of the lower bound is performed is studied. It is shown to be of a multiplicative factor at most equal to $1-n^{\delta}/(n+n^{\delta})(1-\theta)$, where $0 < \theta < 1$ is a fixed parameter in the algorithm. It is deduced from it that the total number of updates of the lower bound in the weighted projective algorithm is at most $O(n^{1-\delta}L)$ for $0 \leq \delta \leq 1$.

Key words. linear programming, interior point method, projective algorithm, path following

AMS subject classification. 90C05

1. Introduction.

In 1984 Karmarkar [13] proposed the first polynomial-time interior point algorithm for linear programming. de Ghellinck and Vial [3] and Todd and Burrell [23] soon after extended the algorithm to handle the case of an unknown optimal value. These algorithms converge in $O(nL)$ iterations, where n is the number of variables and L represents the input size of the problem. Later, Renegar [18] proposed a new interior point method, with a better convergence bound of $O(\sqrt{n}L)$ iterations. His algorithm was named path following because it closely approximates the so-called trajectory of analytic centers or, in short, trajectory of centers. Sonnevend [22] first introduced the notion of analytic center. Megiddo [15] studied the trajectory of centers.

A great number of alternative interior point methods have subsequently appeared in the literature [1], [5], [11], [25], [27]. Those with proved convergence rate of $O(\sqrt{n}L)$ iterations are of the path-following type. The iterates are forced to stay close to the trajectory of centers at all times or to at least regularly visit the vicinity of this trajectory. Path-following methods are commonly derived from a suitable parametrization of the trajectory of centers. Gonzaga [10] and Roos and Vial [20] analyzed the relationships between these parameters. The parameter can be an upper bound for the optimal value (if the linear programming problem of interest is to be minimized) as in [18], a lower bound as in Gonzaga [12], or the penalty term in the logarithmic barrier function (see [10], Goldfarb and Liu [9], Roos and Vial [21] in a primal framework and Kojima et al. [14] and Monteiro and Adler [17] in a primal-dual setting). All these methods may be viewed as operating in an affine space. It appears that when the iterate is "close" to the trajectory of centers it is possible to update the currently used parameter in such a way that one step, or at most a few steps, in the space of the variables brings the iterate back into the vicinity of the trajectory. Though quite small, the update is still large enough to enforce convergence in $O(\sqrt{n}L)$ iterations.

The projective methods [3] and [23] are also based on a parametrization by a lower bound. The search direction they use has much in common with those used in the path-

* Received by the editors October 15, 1990; accepted for publication (in revised form) December 14, 1992. This research was supported by the Fonds National de la Recherche Scientifique Suisse (grant 12-26434.89), the Natural Sciences and Engineering Research Council of Canada, and the Formation de Chercheurs et Action de Recherche, Quebec.

† Groupe d'Etudes et de Recherche en Analyse des Décisions and Faculty of Management, McGill University, Montreal, Quebec, Canada H3A 165.

‡ Département d'Économie Commerciale et Industrielle, 2, Rue de Candolle, CH-1211 Genève 4, Switzerland.

following methods (see the survey paper of den Hertog and Roos [4]). However, there are significant differences in the approaches. In the standard projective methods the parameter is updated much before the iterate gets close to the trajectory. Moreover, the updating is significantly deeper. However, by getting away from the trajectory of centers it seems that one loses some kind of control on the convergence analysis and one can only prove convergence in $O(nL)$ iterations. Goffin and Vial [8] proposed a weighted version of the standard projective algorithm with the same convergence result of $O(nL)$ iterations.

The first short-step projective algorithm is due to Roos [19]. His method applies to the dual of Karmarkar's canonical program. Todd and Ye [24] also devised a path-following method in a projective framework. They introduced a new primal-dual potential function and used it to measure progress towards optimality. Their path-following algorithm could be viewed as a parameter-free variant of [14] or [17]. Ye [28] and Freund [6] use the potential of Todd and Ye [24] with weight $n+\sqrt{n}$ on the first term to build an algorithm in the affine framework. Although their method is based on a parameter-free potential, it can properly be seen as being parametrized by a lower bound. This bound is the dual objective function value associated with the dual variables that are generated as soon as the iterate is close to the trajectory of centers. However, the distinguished feature of their method is that it allows a line search on those dual variables. Mizuno [16] discussed variants of this algorithm, also in the affine framework, with a more general weight on the first term of the potential.

In [29] Ye devised a primal projective algorithm that uses the primal potential with weight $n+\lceil\sqrt{n}\rceil+1$. A centering condition based on the norm of the scaled steepest descent direction of the objective is used. The dual bound is updated by a line search in the space of the dual variables, using the primal-dual potential function. The iteration count $O(\sqrt{n}L)$ is obtained. More recently, Xiao and Goldfarb [26] studied a path-following variant of Karmarkar's projective algorithm. The steps are essentially performed in the primal space but, by staying close to the trajectory of centers, they generate in the meantime dual variables that are used to measure progress towards optimality through the potential function. Anstreicher [2] gave a monotonic variant that allows line searches.

In this paper we also present a primal projective short-step algorithm based on a weighted potential of the Karmarkar type. The use of the variant of Karmarkar's algorithm of [3] permits a centrality measure based on the norm of the Newton direction and properly interprets the algorithm as generating a sequence of iterates (primal variables and dual bounds) that remain in the radius of quadratic convergence of Newton's method. If the weight on the first term of the potential is $n+n^{\delta}, \delta \geq 0$, the algorithm converges in $O(n^{\max\{\delta/2,1-\delta/2\}}L)$ iterations. We give two proofs: one is based on duality gap reductions, the other on potential reductions. We also propose a medium-step version of the algorithm. There, the lower bound parameter is updated to the maximum value that is compatible with the path-following constraint of a close approximation of the trajectory of centers at each step of the algorithm. Finally, the same tools are used to bound the number of updates of the lower bound parameter in the weighted projective algorithm of [8]. With a weight $n+n^{\delta}$ on the first term of the potential, this number is $O(n^{1-\delta}L)$ if $0 \leq \delta < 1$ and $O(L)$ if $\delta \geq 1$, whereas, as shown in [8], the total number of iterations is bounded by $O((n+n^{\delta})L)$.

As far as notation is concerned, we use capital letters $X, W, \ldots$, to denote the diagonal matrices whose diagonal elements are the components of the vectors $x, w, \ldots$. On the other hand, $\mathbf{1}$ designates the vector of all ones of appropriate size.

2. Linear programming and weighted analytic centers. We consider the pair of dual linear programming problems:

$$\min\{\tilde{c}^T\tilde{x} : \tilde{A}\tilde{x} = \tilde{b}, \tilde{x} \geq 0\},$$

$$\max\{\tilde{b}^T\tilde{u} : \tilde{A}^T\tilde{u} + \tilde{s} = \tilde{c}, \tilde{s} \geq 0\},$$

where $\tilde{A}$ is an $m \times n$ matrix, $\tilde{b}$ and $\tilde{c}$ are $m \times 1$ and $n \times 1$ columns, respectively, and $\tilde{x}$ and $\tilde{u}$ are n-dimensional and m-dimensional vectors in which the optimizations are done. The vector $\tilde{s} \in \mathbb{R}^n$ is the vector of dual slacks.

To simplify the analysis we shall assume that $\tilde{A}$ has full rank.

We make the usual assumption that the sets of primal and dual interior feasible solutions, respectively,

$$\tilde{F}_P = \{\tilde{x} : \tilde{A}\tilde{x} = \tilde{b}, \tilde{x} > 0\},$$
$$\tilde{F}_D = \{\tilde{u} : \tilde{A}^T\tilde{u} + \tilde{s} = c, \tilde{s} > 0\},$$

are nonempty. We shall denote by z^* the optimal value of the problems.

Let $\nu \geq 1$ and let z_ℓ be a strict lower bound on the optimum. We define Karmarkar's weighted potential function as

$$\tilde{\varphi}(\tilde{x}; z_\ell) = (n + \nu)\ln(\tilde{c}^T\tilde{x} - z_\ell) - \sum_{j=1}^{n} \ln \tilde{x}_j.$$

$\tilde{\varphi}(\tilde{x}; z_\ell)$ is well defined on $\tilde{F}_P$.

In [8] we introduced the weighted analytic center by

$$\tilde{x}(z_\ell) = \arg\min_{\tilde{x} \in \tilde{F}_P} \tilde{\varphi}(\tilde{x}; z_\ell).$$

$\tilde{x}(z_\ell)$ is the unique solution of the set of equations in $\tilde{x}, \tilde{s}$, and $\tilde{u}$:

$$\tilde{A}\tilde{x} = \tilde{b}, \tag{1}$$

$$\tilde{A}^T\tilde{u} + \tilde{s} = \tilde{c}, \tag{2}$$

$$\tilde{S}\tilde{x} = \frac{(\tilde{c}^T\tilde{x} - z_\ell)}{n + \nu}\mathbf{1}. \tag{3}$$

It is easy to verify from the last equation that $\tilde{s} > 0$ and hence $\tilde{u} \in \tilde{F}_D$. Furthermore,

$$\tilde{c}^T\tilde{x} - \tilde{b}^T\tilde{u} = \tilde{s}^T\tilde{x} = \frac{\tilde{c}^T\tilde{x} - z_\ell}{n + \nu}\mathbf{1}^T\mathbf{1} = (\tilde{c}^T\tilde{x} - z_\ell)\frac{n}{n + \nu}. \tag{4}$$

Equation (4) provides a lower bound $\tilde{b}^T\tilde{u}$ that reduces the duality gap $\tilde{c}^T\tilde{x} - z_\ell$ by a multiplicative factor $n/(n + \nu)$ independent of $\tilde{x}$.

3. Primal projective search direction and the short-step algorithm. In this section we present the weighted projective algorithm of [8], which is based on the minimization of the weighted potential introduced in §2. This algorithm is an extension of the algorithm of [3]. The purpose of the present section is to give a short-step version of this algorithm.

We first review the definition and the motivation of the primal search direction in [8]. In this paper the original problem is embedded in a projective space by adding an auxiliary (homogenizing) variable $x_{n+1} > 0$. Introducing the notation

$$x = \begin{pmatrix} \tilde{x} \\ x_{n+1} \end{pmatrix}, \quad A = (\tilde{A} \quad -\tilde{b}), \quad c(z_\ell) = \begin{pmatrix} \tilde{c} \\ -z_\ell \end{pmatrix},$$

$$F_P = \{x > 0 : Ax = 0\},$$

and

$$\varphi(x, z_\ell) = (n+\nu)\ln(c(z_\ell)^T x) = \sum_{j=1}^{n+1} \ln x_j,$$

we see that for any $x \in F_P$, the first n coordinates of x/x_{n+1} uniquely define a point $\tilde{x} \in \tilde{F}_P$. Moreover, φ being positively homogeneous of degree zero,

$$\varphi(x; z_\ell) = \varphi\left(\frac{x}{x_{n+1}}; z_\ell\right) = \tilde{\varphi}(\tilde{x}; z_\ell).$$

Hence, minimizing φ with respect to x over the set F_P is equivalent to minimizing $\tilde{\varphi}(\tilde{x}; z_\ell)$ over $\tilde{F}_P$. By the same token, it is always possible to scale $x > 0$ to have $x_{n+1} = 1$.

Finally, we define the $(m+1) \times (n+1)$ and $(n+1) \times (n+1)$ matrices D and N by

$$D = \begin{pmatrix} \tilde{c}^T & -z_\ell \\ \tilde{A} & -\tilde{b} \end{pmatrix}$$

and

$$N = \begin{pmatrix} I & 0 \\ 0 & \sqrt{\nu} \end{pmatrix}.$$

The search direction p for the modified projective algorithm is the (unique) solution of the weighted least squares set of equations in the unknowns p and v

$$N^2 X^{-2} p - N^2 X^{-1}\mathbf{1} + D^T v = 0, \tag{5}$$

$$Dp = 0. \tag{6}$$

This direction is a sensible choice because it maintains feasibility and ensures that the nonconvex first term of $\varphi(x, z_\ell)$ remains constant in the null space $\{p : Dp = 0\}$. Thus, φ is strongly convex in that subspace and p can be interpreted as a projected Newton direction with respect to φ. This interpretation is not essential for the rest of the paper but we shall hereafter refer to p as the projected Newton direction.

It is convenient to introduce the vectors $q = NX^{-1}p$ and $t = N^{-1}XD^T v$. The vector q is the scaled projected Newton direction. We then write (5) and (6) as

$$N\mathbf{1} = q + t, \tag{7}$$

$$(DN^{-1}X)q = 0 \quad \text{and} \quad t = (DN^{-1}X)^T v. \tag{8}$$

q and t are thus the elements of the orthogonal decomposition of the vector $N\mathbf{1}$, respectively, on the null space of $(DN^{-1}X)$ and on the range space of $(DN^{-1}X)^T$. It is possible to give an analytical expression of q (see for instance Goffin and Vial [8]). We shall denote it $q(z_\ell)$ to emphasize its dependence on z_ℓ. Then

$$\begin{aligned} q(z_\ell) &= N\mathbf{1} - N^{-1}XD^T v \\ &= N\mathbf{1} - \frac{c(z_\ell)^T x}{\|N^{-1}Xs(z_\ell)\|^2} N^{-1}Xs(z_\ell), \end{aligned} \tag{9}$$

with

$$s(z_\ell) = c(z_\ell) - A^T u(z_\ell) \tag{10}$$

and

$$u(z_\ell) = (AX^2N^{-2}A^T)^{-1}AX^2N^{-2}c(z_\ell). \tag{11}$$

Note that q is positively homogeneous of degree 0 with respect to x. We can always assume, as will be done in the algorithm, that q is computed at points that are scaled to $x_{n+1} = 1$. In that case

$$c(z_\ell)^T x = \tilde{c}^T\tilde{x} - z_\ell,$$

an expression we shall use in the sequel without justifying it again. Note that the fact that z_ℓ is a lower bound is not essential in the definition of q, as given by (9)–(11). However, φ is well defined only if z_ℓ is a strict lower bound.

With these definitions in mind one can write the projected Newton step as

$$\bar{x} = X(\mathbf{1} + N^{-1}q(z_\ell)). \tag{12}$$

We can now formally state the short-step algorithm.

SHORT-STEP PRIMAL PROJECTIVE ALGORITHM
Data: $\varepsilon = 2^{-L}; 0 < \theta < 1; \sigma = \nu^{\frac{1}{2}}(n+\nu)^{-1}\min\{(1-\theta)/8, \sqrt{\theta(1-\theta)}/2\}; x > 0$, with $Ax = 0; z_\ell$ a lower bound and $\|q(z_\ell)\| \le \theta$.
Basic step: given $x = \binom{\tilde{x}}{1}$ and z_ℓ

begin
  while $\tilde{c}^T\tilde{x} - z_\ell > \varepsilon$ **do**
  begin
    compute $q(\cdot)$ by (9)
    $z_\ell := z_\ell + \sigma(\tilde{c}^T\tilde{x} - z_\ell)$
    $\bar{x} := x + N^{-1}Xq(z_\ell)$
    $x := \frac{1}{\bar{x}_{n+1}}\bar{x}$
  end
end

The value σ plays an important role in the progress of the algorithm. The larger σ is, the bigger is the increase of the lower bound z_ℓ. The default value

$$\sigma = \nu^{\frac{1}{2}}(n+\nu)^{-1}\min\left\{\frac{1-\theta}{8}, \frac{\sqrt{\theta(1-\theta)}}{2}\right\}$$

ensures that $\bar{z} = z_\ell + \sigma(\tilde{c}^T\tilde{x} - z_\ell)$ is a lower bound and that, after the updating of x and z_ℓ, $\|\bar{q}(\bar{z})\| \le \theta$. We shall prove in the next section that

$$\hat{z} = \arg\max\{z : \tilde{c}^T\tilde{x} - z \ge 0 \text{ and } \|q(z)\| \le \sqrt{\theta}\}$$

meets the same conditions but corresponds to a larger value of $\hat{z} \ge \bar{z}$. We call the following version of the algorithm medium step.

MEDIUM-STEP PRIMAL PROJECTIVE ALGORITHM
Data: $\varepsilon = 2^{-L}; 0 < \theta < 1; x > 0$, with $Ax = 0; z_\ell$ a lower bound and $\|q(z_\ell)\| \le \theta$.
Basic step: given $x = \binom{\tilde{x}}{1}$ and z_ℓ

begin
  while $\tilde{c}^T\tilde{x} - z_\ell > \varepsilon$ **do**

begin
compute $q(\cdot)$ by (9) at x
$z_\ell := \arg\max\{z : \tilde{c}^T\tilde{x} - z \geq 0 \text{ and } \|q(z)\| \leq \sqrt{\theta}\}$
$\bar{x} := x + N^{-1}Xq(z_\ell)$
$x := \frac{1}{\bar{x}_{n+1}}\bar{x}$
end

4. Convergence analysis. In this section, we shall implicitly assume, without loss of generality, that $x_{n+1} = 1$. We note first that by definition $p(z_\ell) = N^{-1}Xq(z_\ell)$ is in the null space of A. Then

$$A\bar{x} = A\big(x + p(z_\ell)\big) = Ax = 0.$$

If, furthermore, $\bar{x} \geq 0$, then $\bar{x}$ is primal feasible. By (12), $\|q(z_\ell)\| < 1$ is a sufficient condition for $\bar{x} > 0$. This result can be sharpened to the following lemma.

LEMMA 1. *Assume* $\tilde{c}^T\tilde{x} - z_\ell \geq 0$. *If* $\|q(z_\ell)\| < 1$, *then* $\bar{x}$ *and* $u(z_\ell)$ *are interior primal and dual feasible solutions, respectively.*

Proof. Since $\|q(z_\ell)\| < 1$ implies $q(z_\ell) < \mathbf{1} \leq N\mathbf{1}$, the conclusion follows directly from (9) and the condition $\tilde{c}^T\tilde{x} - z_\ell \geq 0$. □

LEMMA 2. $\|q(z_\ell)\| \leq \|N\mathbf{1} - N^{-1}XD^Tv\|$ *for all* $v \in \mathbb{R}^{m+1}$.

Proof. $q(z_\ell)$ is the projection of $N\mathbf{1}$ over the null space of DXN^{-1} and is thus the vector of smallest norm having the form $N\mathbf{1} - N^{-1}XD^Tv$. □

LEMMA 3. *Assume* $\tilde{c}^T\tilde{x} - z_\ell > 0$. *If* $\|q(z_\ell)\| \leq \theta$,

$$\tilde{b}^Tu(z_\ell) - z_\ell \geq (\tilde{c}^T\tilde{x} - z_\ell)\frac{\nu - \theta\sqrt{\nu}}{n + \nu}.$$

Moreover, if $\theta < 1$, *then* $z^* > \tilde{b}^Tu(z_\ell) > z_\ell$.

Proof. From the hypothesis one has $q_{n+1}(z_\ell) \leq \theta$. Since

$$q_{n+1}(z_\ell) = \sqrt{\nu} - \frac{1}{\sqrt{\nu}}\frac{\tilde{c}^T\tilde{x} - z_\ell}{\|N^{-1}Xs(z_\ell)\|^2}x_{n+1}s_{n+1}(z_\ell),$$

and $x_{n+1} = 1$ and $s_{n+1}(z_\ell) = \tilde{b}^Tu(z_\ell) - z_\ell$, then

$$\theta \geq q_{n+1}(z_\ell) = \frac{1}{\sqrt{\nu}}\left(\nu - \frac{\tilde{c}^T\tilde{x} - z_\ell}{\|N^{-1}Xs(z_\ell)\|^2}(\tilde{b}^Tu(z_\ell) - z_\ell)\right).$$

Since $\tilde{c}^T\tilde{x} - z_\ell > 0$,

$$\tilde{b}^Tu(z_\ell) - z_\ell \geq (\nu - \theta\sqrt{\nu})\frac{\|N^{-1}Xs(z_\ell)\|^2}{\tilde{c}^T\tilde{x} - z_\ell}. \tag{13}$$

From (7), the orthogonal decomposition of $N\mathbf{1}$ implies

$$n + \nu = \|N\mathbf{1}\|^2 = \|q(z_\ell)\|^2 + \frac{(\tilde{c}^T\tilde{x} - z_\ell)^2}{\|N^{-1}Xs(z_\ell)\|^2}. \tag{14}$$

Thus, by (13) and (14) and $\|q(z_\ell)\| \leq \theta$

$$\tilde{b}^Tu(z_\ell) - z_\ell \geq \frac{\nu - \theta\sqrt{\nu}}{n + \nu}(\tilde{c}^T\tilde{x} - z_\ell).$$

By Lemma 1, $u(z_\ell)$ is interior feasible if $\theta < 1$. Moreover, $\theta < 1$ also implies that $(\nu - \theta\sqrt{\nu})/(n+\nu) > 0$. Hence, the conclusion. □

The next lemma was proved in [8]. For the sake of completeness, we give here a separate proof of it.

LEMMA 4. *If* $\bar{x} \geq 0$,

$$\|\bar{q}(z_\ell)\| \leq \|q(z_\ell)\|^2,$$

where $\bar{q}(z_\ell)$ *is computed at* $\bar{x} = X(\mathbf{1} + N^{-1}q(z_\ell))$.

Proof. Let v and $\bar{v}$ be the vectors that enter the definition (9) of $q(z_\ell)$ and $\bar{q}(z_\ell)$, respectively. From Lemma 1

$$\|\bar{q}(z_\ell)\| = \|N\mathbf{1} - N^{-1}\bar{X}D^T\bar{v}\| \leq \|N\mathbf{1} - N^{-1}\bar{X}D^T v\|.$$

From the definition of $\bar{x}$

$$\begin{aligned}\bar{x} &= X(\mathbf{1} + N^{-1}(N\mathbf{1} - N^{-1}XD^T v)) \\ &= (2X - N^{-2}X^2W)\mathbf{1},\end{aligned}$$

where $W = \operatorname{diag}(D^T v)$ is the $(n+1) \times (n+1)$ diagonal matrix having the components of the vector $D^T v$ on its main diagonal. One can write

$$\begin{aligned}N\mathbf{1} - N^{-1}\bar{X}D^T v &= N(I - N^{-2}\bar{X}W)\mathbf{1} \\ &= N(I - 2N^{-2}XW + N^{-4}X^2W^2)\mathbf{1} \\ &= N^{-1}(N - N^{-1}XW)^2\mathbf{1}.\end{aligned}$$

Let $F = (N - N^{-1}XW)$. F is diagonal with elements $(f_j)_{j=1\cdots(n+1)}$. Hence,

$$\|F^2\mathbf{1}\|^2 = \sum_{j=1}^{n+1} f_j^4 \leq \left(\sum_{j=1}^{n+1} f_j^2\right)^2 = \|F\mathbf{1}\|^4.$$

On the other hand $\nu_\nu \geq 1$ implies that $\|N^{-1}\| \leq 1$. It follows that

$$\begin{aligned}\|\bar{q}(z_\ell)\|^2 &\leq \|N^{-1}\|^2\|(N - N^{-1}XW)^2\mathbf{1}\|^2 \\ &\leq \|(N - N^{-1}XW)\mathbf{1}\|^4 = \|q(z_\ell)\|^4.\end{aligned}$$ □

LEMMA 5. *Let* $\theta \in (0,1)$ *and* $\sigma \leq \nu^{1/2}(n+\nu)^{-1}\min\{(1-\theta)/8, \sqrt{\theta(1-\theta)}/2\}$. *If* $\|q(z)\| \leq \theta$ *and* $\hat{z} = z + \sigma(\tilde{c}^T\tilde{x} - z)$, *then* $\|q(\hat{z})\| \leq \sqrt{\theta}$, *and* $\hat{z}$ *is a valid lower bound.*

Proof. Let us first introduce some notation. Let $\bar{g} = \binom{0}{1}$ be the $(n+1)$st unit vector and $\bar{c} = \binom{\tilde{c}}{0}$. We denote γ_c and γ_g the projections of $N^{-1}X\bar{c}$ and $N^{-1}X\bar{g}$, respectively, on the null space of $AN^{-1}X$. Thus $\gamma(z) = \gamma_c - z\gamma_g$ is the projection of $N^{-1}Xc(z) = N^{-1}X(\bar{c} - z\bar{g})$ on the null space of $AN^{-1}X$. From (10) and (11), $\gamma(z) = XN^{-1}s(z)$. Finally, let $\Delta(z) = \tilde{c}^T\tilde{x} - z$.

With these definitions in mind, we can write the orthogonal decomposition of $N\mathbf{1}$ as

$$N\mathbf{1} = q(z) + \frac{\Delta(z)}{\|\gamma(z)\|^2}\gamma(z). \tag{15}$$

We now relate $\|q(\hat{z})\|$ to $\|q(z)\|$. Since $\hat{z} = z + \sigma(\tilde{c}^T\tilde{x} - z)$, one has $\Delta(\hat{z}) = (1-\sigma)\Delta(z)$ and $\gamma(\hat{z}) = \gamma_c - z\gamma_g - \sigma\Delta(z)\gamma_g = \gamma(z) - \sigma\Delta(z)\gamma_g$. Thus, in view of the decomposition (15),

$$
\begin{aligned}
\|q(\hat{z})\|^2 &= n+\nu-\frac{\Delta^2(\hat{z})}{\|\gamma(\hat{z})\|^2} \\
&= n+\nu-(1-\sigma)^2\frac{\Delta^2(z)}{\|\gamma(z)\|^2}\frac{\|\gamma(z)\|^2}{\|\gamma(\hat{z})\|^2} \\
&= n+\nu-(1-\sigma)^2(n+\nu-\|q(z)\|^2)\frac{\|\gamma(z)\|^2}{\|\gamma(\hat{z})\|^2}.
\end{aligned} \tag{16}
$$

We shall now bound the term $\|\gamma(z)\|^2/\|\gamma(\hat{z})\|^2$. Simple algebra yields

$$
\frac{\|\gamma(\hat{z})\|^2}{\|\gamma(z)\|^2} = 1-2\sigma\frac{\Delta(z)}{\|\gamma(z)\|^2}\gamma(z)^T\gamma_g+\sigma^2\frac{\Delta^2(z)\|\gamma_g\|^2}{\|\gamma(z)\|^2}. \tag{17}
$$

Since γ_g is the projection of $N^{-1}X\bar{g}$ on the null space of $AN^{-1}X$,

$$
\|\gamma_g\|^2 \le \|N^{-1}X\bar{g}\|^2 = \nu^{-1}. \tag{18}
$$

On the other hand, γ_c and γ_g are both projections on the same null space; thus

$$
\gamma_g^T\gamma(z) = (N^{-1}X\bar{g})^T\gamma(z) = \nu^{-\frac{1}{2}}\gamma_{n+1}(z). \tag{19}
$$

Moreover, the inequality

$$
\theta \ge \|q(z)\| = \left\| N\mathbf{1}-\frac{\Delta(z)}{\|\gamma(z)\|^2}\gamma(z) \right\|
$$

implies that

$$
\nu^{\frac{1}{2}}-\theta \le \frac{\Delta(z)}{\|\gamma(z)\|^2}\gamma_{n+1}(z) \le \nu^{\frac{1}{2}}+\theta.
$$

Thus, by (19)

$$
1-\theta\nu^{-\frac{1}{2}} \le \frac{\Delta(z)}{\|\gamma(z)\|^2}\gamma(z)^T\gamma_g \le 1+\theta\nu^{-\frac{1}{2}}. \tag{20}
$$

Since by (15)

$$
n+\nu = \|q(z)\|^2+\frac{\Delta^2(z)}{\|\gamma(z)\|^2} \ge \frac{\Delta^2(z)}{\|\gamma(z)\|^2},
$$

we can derive from (18), (19), and (20) the bound:

$$
\frac{\|\gamma(\hat{z})\|^2}{\|\gamma(z)\|^2} \le 1-2\sigma(1-\theta\nu^{-\frac{1}{2}})+\sigma^2(n+\nu)\nu^{-1}. \tag{21}
$$

Combining (16) and (21) yields

$$
\begin{aligned}
\|q(\hat{z})\|^2 &\le n+\nu-\frac{(1-\sigma)^2(n+\nu-\theta^2)}{1-2\sigma(1-\theta\nu^{-\frac{1}{2}})+\sigma^2(n+\nu)\nu^{-1}} \\
&= n+\nu-\frac{(n+\nu-\theta^2)}{1+\dfrac{2\sigma\theta\nu^{-\frac{1}{2}}+\sigma^2\nu^{-1}n}{(1-\sigma)^2}}.
\end{aligned} \tag{22}
$$

Thus $\|q(\hat{z})\|^2 \leq \theta$ holds if

$$\frac{2\sigma\theta\nu^{-\frac{1}{2}} + \sigma^2\nu^{-1}n}{(1-\sigma)^2} \leq \frac{\theta - \theta^2}{n+\nu-\theta}$$

or, equivalently, if

$$2\sigma\theta\nu^{-\frac{1}{2}} + \sigma^2\nu^{-1}n \leq (1-\sigma)^2\frac{\theta-\theta^2}{n+\nu-\theta}. \tag{23}$$

Let us temporarily assume that $(1-\sigma)^2 \geq \frac{1}{2}$. Since $0 \leq n+\nu-\theta \leq n+\nu$, (23) holds if

$$2\sigma\theta\nu^{-\frac{1}{2}} + \sigma^2\nu^{-1}n \leq \frac{\theta-\theta^2}{2(n+\nu)}. \tag{24}$$

This inequality is implied in turn by

$$\max\{2\sigma\theta\nu^{-\frac{1}{2}}, \sigma^2\nu^{-1}n\} \leq \frac{\theta-\theta^2}{4(n+\nu)}$$

or, equivalently, by

$$\sigma \leq \nu^{\frac{1}{2}}(n+\nu)^{-1}\min\left\{\frac{1-\theta}{8}, \frac{\sqrt{\theta(1-\theta)}}{2(1-\nu(n+\nu)^{-1})^{\frac{1}{2}}}\right\}. \tag{25}$$

Since $0 < \nu$, then $0 < 1-(n+\nu)^{-1}\nu < 1$ and (25) can be strengthened to

$$\sigma \leq \nu^{\frac{1}{2}}(n+\nu)^{-1}\min\left\{\frac{1-\theta}{8}, \frac{\sqrt{\theta(1-\theta)}}{2}\right\}. \tag{26}$$

This clearly implies that $\sigma \leq \frac{1}{8}$; therefore, $(1-\sigma)^2 \geq \frac{1}{2}$ and the lemma is proved. □

The next lemma asserts that the conditions on z_ℓ and $q(z_\ell)$ remain satisfied after each step of the algorithm.

LEMMA 6. *Assume* $\theta < 1$ *and* $0 \leq \sigma \leq \nu^{1/2}(n+\nu)^{-1}\min\{(1-\theta)/8, \sqrt{\theta(1-\theta)}/2\}$. *Let* $\bar{x} = x + XN^{-1}q(\bar{z})$, *with* $\bar{z} = z_\ell + \sigma(\tilde{c}^T\tilde{x} - z_\ell)$. *If* z_ℓ *is a strict lower bound and* $\|q(z_\ell)\| \leq \theta$, *then* $\bar{z}$ *is a strict lower bound and* $\|\bar{q}(\bar{z})\| \leq \theta$.

Proof. By Lemma 5

$$\|q(\bar{z})\| \leq \sqrt{\theta}.$$

By Lemma 3, since $\tilde{c}^T\tilde{x} - \bar{z} = (1-\sigma)(\tilde{c}^T\tilde{x} - z_\ell) > 0$ and $\|q(\bar{z})\| < 1$, then $\bar{z}$ is a lower bound. By Lemma 1, $\bar{x}$ is an interior feasible solution. Finally, by Lemma 4, $\|\bar{q}(\bar{z})\| \leq \|q(\bar{z})\|^2 \leq \theta$. □

THEOREM 7. *Assume the following hypotheses*:

(i) x^0 *is an interior feasible point with* $x_j^0 > 2^{-L}, j = 1, \ldots, n$,

(ii) $z_\ell^0 < z^*$ *is a lower bound such that* $\tilde{c}^T\tilde{x}^0 - z_\ell^0 \leq 2^L$,

(iii) $\|q(z_\ell^0)\| \leq \theta < 1$,

(iv) $\sigma = \nu^{1/2}(n+\nu)^{-1}\min\{(1-\theta)/8, \sqrt{\theta(1-\theta)}/2\}$.

After at most $K = O(\frac{L}{\sigma})$ *iterations of the short-step projective algorithm,* $\tilde{c}^T\tilde{x}^K - z_\ell^K < 2^{-L}$. *Moreover, if* $\nu = n^\delta$, *then* $K = O(n^{\max\{\delta/2, 1-\delta/2\}}L)$.

Remarks. (1) Condition (iii) implies that there exists a dual interior feasible solution. Hence the optimal set of the primal problem is bounded. This is the classical condition for convergence of most interior point methods.

(2) The best convergence result is $O(\sqrt{n}L)$ for $\delta = 1$, i.e., $\nu = n$.

We shall give two proofs of this theorem. One is based on duality gap reductions, the other on potential reductions.

A first proof of Theorem 7. The hypotheses of Lemma 6 are satisfied. Hence, for all k, $\|q(z_\ell^k)\| \leq \theta$ and $z_\ell^k \leq z^*$. The algorithm is well defined. Moreover, at iteration k

$$z^* - z_\ell^k \leq z^* - z_\ell^{k-1} - \sigma(\tilde{c}^T\tilde{x}^{k-1} - z_\ell^{k-1}) \leq (1-\sigma)(z^* - z_\ell^{k-1}).$$

Hence,

$$z^* - z_\ell^k \leq (1-\sigma)^k(z^* - z_\ell^0) \leq (1-\sigma)^k(\tilde{c}^T\tilde{x}^0 - z_\ell^0).$$

By Lemma 3

$$\begin{aligned}\tilde{c}^T\tilde{x}^k - z_\ell^k &\leq \frac{n+\nu}{\nu - \theta\sqrt{\nu}}(\tilde{b}^T u^k(z_\ell^k) - z_\ell^k)\\ &\leq \frac{n+\nu}{\nu - \theta\sqrt{\nu}}(z^* - z_\ell^k)\\ &\leq \frac{n+\nu}{\nu - \theta\sqrt{\nu}}(1-\sigma)^k(\tilde{c}^T\tilde{x}^0 - z_\ell^0).\end{aligned}$$

Taking the logarithm on both sides of the inequality and using the obvious inequality $\log(1+t) \leq t$ for all $t > -1$, one gets that $\tilde{c}^T\tilde{x}^k - z_\ell^k > 2^{-L}$ implies

$$k \leq 2\frac{L}{\sigma} + \frac{1}{\sigma}\ln\left(1 + \frac{n}{\nu}\right) \leq \frac{1}{\sigma}\left(2L + \frac{n}{\nu}\right) \leq \frac{3L}{\sigma}.$$

The last inequality follows from $n < L$. Hence, K is at most $O(\frac{L}{\sigma}) = O(n^{\max\{\delta/2, 1-\delta/2\}}L)$.

A second proof of Theorem 7. This proof holds only for $\theta \leq \frac{1}{2}$. Under the assumptions of the theorem, $\varphi(x^0; z_\ell^0) = O((n+\nu)L)$. Since by (iii) the optimal set is bounded, the condition $\varphi(x^K; z_\ell^K) = O(-(n+\nu)L)$, for some x^K and some z_ℓ^K, is sufficient to guarantee $\tilde{c}^T\tilde{x}^K - z_\ell^K < 2^{-L}$. Let us denote for short $q = q(z_\ell^k)$ the scaled projected Newton step at iteration k computed at $x = x^{k-1}$ and $z = z_\ell^k$. The following inequality is due to Karmarkar. For any $\lambda \in \mathbb{R}$, with $|\lambda| < 1$,

$$\ln(1+\lambda) \geq \lambda - \frac{\lambda^2}{2(1-|\lambda|)}. \tag{27}$$

Let $\lambda_j = q_j, j = 1, \ldots, n$, and let $\lambda_{n+1} = \nu^{-1/2}q_{n+1}$. Since $\|q\| \leq \theta < 1$, then $|\lambda_j| < 1$, $j = 1, \ldots, n+1$ and we can apply inequality (27) for each λ_j. Upon summing these

inequalities, one gets

$$
\begin{aligned}
\varphi(x^k; z_\ell^k) - \varphi(x^{k-1}; z_\ell^k) &= -\sum_{j=1}^{n} \ln(1+q_j) - \nu \ln(1+\nu^{\frac{1}{2}} q_{n+1}) \\
&\le -\sum_{j=1}^{n} \left(q_j - \frac{q_j^2}{2(1-|q_j|)} \right) \\
&\quad -\nu \left(\nu^{-\frac{1}{2}} q_{n+1} - \frac{\nu^{-1} q_{n+1}^2}{2(1-\nu^{-\frac{1}{2}}|q_{n+1}|)} \right) \\
&\le -\mathbf{1}^T N q + \frac{\|q\|^2}{2(1-\theta)} = \|q\|^2 \frac{2\theta - 1}{2(1-\theta)} \le 0.
\end{aligned} \tag{28}
$$

On the other hand,

$$
\varphi(x^{k-1}; z_\ell^k) - \varphi(x^{k-1}; z_\ell^{k-1}) = (n+\nu)\ln(1-\sigma) \le -\sigma(n+\nu). \tag{29}
$$

Combining (28) and (29) we get

$$
\varphi(x^k; z_\ell^k) - \varphi(x^{k-1}; z_\ell^{k-1}) \le -\sigma(n+\nu).
$$

Thus, after at most $K = O\left(\frac{L}{\sigma}\right)$ iterations, the algorithm stops with $\varphi(x^K; z_\ell^K) = O(-(n+\nu)L)$. □

The assumptions (i)–(iii) of Theorem 7 are not restrictive. Given any linear programming problem (with integer data), it is possible to transform it in polynomial time into an equivalent problem of comparable size for which there is a known interior point solution satisfying the hypotheses (i)–(iii). For a justification of this result we refer to Renegar [18] or Monteiro and Adler [17].

A similar result holds for the medium-step projective algorithm.

THEOREM 8. *Assume hypotheses* (i), (ii), *and* (iii) *of Theorem* 7 *and let* $\nu = n^\delta$, *for some* $\delta > 0$. *After at most* $K = O(n^{\max\{\delta/2, 1-\delta/2\}} L)$ *iterations of the medium-step projective algorithm,* $\tilde{c}^T \tilde{x}^K - z_\ell^K < 2^{-L}$.

Proof. Assume that z_ℓ is the current value of the parameter with $\|q(z_\ell)\| \le \theta$. The new value of the parameter is defined by

$$
\bar{z} = \arg\max\{z : \tilde{c}^T \tilde{x} - z \ge 0 \text{ and } \|q(z)\| \le \sqrt{\theta}\}.
$$

We claim that this maximization is well posed and that its solution $\bar{z}$ is a lower bound. Note that we can add the constraint $z \ge z_\ell$ to the maximization problem without altering its solution. It follows then from the continuity of the function $\|q(\cdot)\|$ that the feasible set of the maximization problem is compact. Hence, the problem is well posed. Since $\|q(\tilde{c}^T \tilde{x})\| = \|N\mathbf{1}\| = \sqrt{n+\nu} > 1 > \sqrt{\theta}$ for $z = \tilde{c}^T \tilde{x}$, then necessarily $\tilde{c}^T \tilde{x} - \bar{z} > 0$ and $\|q(\bar{z})\| = \sqrt{\theta}$. Thus, by Lemma 3, $\bar{z}$ is a strict lower bound. The claim is proved.

To complete the proof of the theorem it remains to show that the medium-step algorithm achieves at each iteration a duality gap reduction at least as large as in the short-step algorithm. Let $\hat{z} = z_\ell + \sigma(\tilde{c}^T \tilde{x} - z_\ell)$ be the updated value of the lower bound in the short-step algorithm. By construction $\hat{z} \le \tilde{c}^T \tilde{x}$ and $\|q(\hat{z})\| \le \sqrt{\theta}$. Hence, $\bar{z} \ge \hat{z}$. The claim and the theorem are proved. □

Remarks. (1) The computation of $\bar{z}$ in the medium-step algorithm is simple. From the orthogonal decomposition in (7), one sees that the equation $\|q(z)\| = \sqrt{\theta}$ is equivalent to

$$
\frac{(\tilde{c}^T \tilde{x} - z)^2}{\|N^{-1} X s(z)\|^2} = n + \nu - \theta.
$$

Since $s(z)$ is linear, the equation is quadratic with a unique root in the interval $[z_\ell, \tilde{c}^T\tilde{x})$.

(2) The medium-step algorithm performs a stronger updating of the lower bound. Though attractive in practice, this property does not materialize in an improvement of the complexity bound.

5. Duality gap in Karmarkar's algorithm. It is proved in [8] that the weighted projective algorithm converges in $O((n+\nu)L)$ iterations. In this section we want to compare the weighted projective algorithm with potential

$$\varphi(x; z_\ell) = (n+\nu)\ln(c(z_\ell))^T x - \sum_{j=1}^{n} \ln x_j$$

and the short-step projective algorithm of §3.

The two algorithms use the same search direction $q(z_\ell)$. They differ essentially in the length of the step that is taken along $q(z_\ell)$ and in the way z_ℓ is updated. In the short-step algorithm the step length is always 1 and z_ℓ is updated at each iteration by a fixed fraction σ of the duality gap $\tilde{c}^T\tilde{x} - z_\ell$. In the weighted projective algorithm of [8] the step length along the search direction is designed so as to (approximately) minimize the weighted potential. The bound z_ℓ is updated as soon as $q(z_\ell) < N\mathbf{1}$, a weaker criterion than $\|q(z_\ell)\| < \theta < 1$.

Let us briefly recall the updating mechanism. From (9)–(11), $q(z_\ell) < N\mathbf{1}$ implies that $s(z_\ell) > 0$. Hence, $u(z_\ell)$, as defined in (11), is dual feasible. Moreover, from $s_{n+1} > 0$, then $\tilde{b}^T u(z_\ell) - z_\ell > 0$. From (11), $u(z_\ell)$ is a linear function of z_ℓ. The largest value z for which $s(z) \geq 0$ still holds ensures that $u(z)$ is dual feasible and, from $\tilde{b}^T u(z) - z = s_{n+1}(z) \geq 0$, that z is a lower bound. Moreover, after this updating $q(z) \not< N\mathbf{1}$.

A stronger updating rule was proposed in [3]. It is defined by

$$\bar{z} = \max\{\tilde{b}^t u(z) : s_j(z) \geq 0, j = 1, \ldots, n\}, \tag{30}$$

where $s(z)$ and $u(z)$ are defined by (10) and (11). This problem can be given a simpler form. Namely

$$\bar{z} = \tilde{b}^T u(\hat{z}), \quad \text{where } \hat{z} = \max\{z : s_j(z) \geq 0, j = 1, \ldots, n\}. \tag{31}$$

The equivalence between (30) and (31) is a straightforward consequence of the following lemma.

LEMMA 9. *Let $u(z)$ be defined by* (11), *with z instead of z_ℓ. Then $\tilde{b}^T u(z)$ is a linear strictly increasing function of z.*

Proof. Because $c(z) = \binom{\tilde{c}}{0} - z\binom{0}{1}$ and $AX^2N^{-2}\binom{0}{1} = -\nu^{-1}\tilde{b}$, then

$$\tilde{b}^T u(z) = (\nu^{-1}\tilde{b}^T(AX^2N^{-2}A^T)^{-1}\tilde{b})z + \tilde{b}^T(AX^2N^{-2}A^T)^{-1}AX^2N^{-2}\begin{pmatrix}\tilde{c}\\0\end{pmatrix}.$$

The lemma follows from $\tilde{b}^T(AX^2N^{-2}A^T)^{-1}\tilde{b} > 0$. □

The updating procedure (31) selects the largest possible lower bound. We name the algorithm max step after this property, to contrast it with the short- and medium-step algorithms of §3.

MAX-STEP PRIMAL PROJECTIVE ALGORITHM
Data: $\varepsilon = 2^{-L}; 0 < \theta < 1; x > 0$, with $Ax = 0$; z_ℓ a lower bound.
Basic step: given $x = \binom{\tilde{x}}{1}$ and z_ℓ
begin
while $\tilde{c}^T\tilde{x} - z_\ell > \varepsilon$ **do**

begin
compute $q(\cdot)$ by (9)
if $q(z_\ell) \leq \theta N\mathbf{1}$
$z_\ell := \tilde{b}^T u(\hat{z})$, where $\hat{z} = \max\{z : s_j(z) \geq 0, j = 1, \ldots, n\}$
update $q(z_\ell)$
endif
$\bar{\alpha} = \arg\min\{\varphi(x + \alpha X N^{-1} q(z_\ell)) : x + \alpha X N^{-1} q(z_\ell) > 0\}$
$\bar{x} := x + \bar{\alpha} N^{-1} X q(z_\ell)$
$x := \frac{1}{\bar{x}_{n+1}} \bar{x}$
end
end

Remark. The standard projective algorithm uses $\theta = 1$.

The convergence analysis of this algorithm is based on the property that each iteration entails a decrease of the potential of at least $O(1)$. One easily concludes that convergence is achieved in $O((n+\nu)L)$ iterations. If $\nu = O(n^\delta)$, this rate is $O(nL)$ for all $0 \leq \delta \leq 1$. For a detailed analysis the reader is referred to [8].

To study the behavior of the duality gap, we use a slightly different version of Lemma 3.

LEMMA 10. *Assume z_ℓ is a lower bound. If $q(z_\ell) \leq \theta N\mathbf{1}$,*

$$\tilde{b}^T u(z_\ell) - z_\ell \geq (\tilde{c}^T \tilde{x} - z_\ell) \frac{\nu(1-\theta)}{n+\nu}.$$

Moreover, if $\theta < 1$ and $z^* > z_\ell$, $u(z_\ell)$ is dual feasible and $\tilde{b}^T u(z_\ell)$ is a lower bound with $z^* \geq \tilde{b}^T u(z_\ell) > z_\ell$.

The proof is essentially the proof of Lemma 3, where the inequality $q_{n+1}(z_\ell) \leq \theta$ is replaced by $q_{n+1}(z_\ell) \leq \theta\sqrt{\nu}$.

THEOREM 11. *Under assumptions* (i), (ii), *and* (iii) *of Theorem* 7, *the number of updates of z_ℓ in the max-step algorithm is at most $K = O\left(\frac{n+\nu}{\nu} L\right)$.*

Proof. Let $\{k_i\}_{i=1,\ldots}$ be the sequence of iterations at which an update of z_ℓ is performed. By Lemma 9, (31) is equivalent to (30) and $z_\ell^{k_i}$ satisfies

$$z_\ell^{k_i} \geq \tilde{b}^T u^{k_i}(z_\ell^{k_i - 1}). \tag{32}$$

By Lemma 10

$$\begin{aligned} \tilde{b}^T u^{k_i}(z_\ell^{k_i-1}) - z_\ell^{k_i-1} &\geq (\tilde{c}^T \tilde{x}^{k_i-1} - z_\ell^{k_i-1}) \frac{\nu(1-\theta)}{n+\nu} \\ &\geq (z^* - z_\ell^{k_i-1}) \frac{\nu(1-\theta)}{n+\nu}. \end{aligned}$$

Hence, by equation (32),

$$\begin{aligned} z^* - z_\ell^{k_i} &\leq z^* - \tilde{b}^T u^{k_i}(z_\ell^{k_i-1}) \\ &\leq (z^* - z_\ell^{k_i-1}) + (z_\ell^{k_i-1} - \tilde{b}^T u^{k_i}(z_\ell^{k_i-1})) \\ &\leq (z^* - z_\ell^{k_i-1}) \left(1 - \frac{\nu(1-\theta)}{n+\nu}\right). \end{aligned}$$

Since $z_\ell^{k_i-1} = z_\ell^{k_{i-1}}$, one concludes, using the argument of the first proof of Theorem 7,

$$\tilde{c}^T \tilde{x}^{k_i-1} - z_\ell^{k_{i-1}} \leq \frac{n+\nu}{\nu(1-\theta)} \left(1 - \frac{\nu(1-\theta)}{n+\nu}\right)^{i-1} (\tilde{c}^T \tilde{x}^0 - z_\ell^0). \tag{33}$$

If at iterate $k_i - 1$,

$$2^{-L} < \tilde{c}^T \tilde{x}^{k_i - 1} - z_\ell^{k_i - 1},$$

then (33) implies

$$2^{-L} \leq \frac{n+\nu}{\nu(1-\theta)} \left(1 - \frac{\nu(1-\theta)}{n+\nu}\right)^{i-1} (\tilde{c}^T \tilde{x}^0 - z_\ell^0).$$

Taking logarithms gives an upper bound on i and therefore proves the theorem. □

If $\nu = 1$, as in Karmarkar's original algorithm, the bound is $O(nL)$. The result is trivial since there is at most one update per iteration and the total number of iterations is $O(nL)$. However, if $\nu = O(n^\delta)$, with $0 \leq \delta < 1$, then the number of updates is bounded by $O(n^{1-\delta}L)$, whereas the total number of iterations is still $O(nL)$. Finally, if $\delta \geq 1$, the total number of updates is $O(L)$.

Acknowledgment. We thank C. Roos for useful discussions on an early draft of the paper, and D. den Hertog for correcting an earlier version of Lemma 5.

REFERENCES

[1] I. Adler and R. D. C. Monteiro, *Limiting behaviour of the affine scaling continuous trajectories for linear programming problems*, Math. Programming, 50 (1991), pp. 29–51.

[2] K. Anstreicher, *Strict monotonicity and improved complexity in the standard form projective algorithm for LP*, CORE discussion paper, Center for Operations Research and Econometrics, Louvain-la-Neuve, Belgium, 1990.

[3] G. de Ghellinck and J.-P. Vial, *A polynomial Newton method for linear programming*, Algorithmica, 1 (1986), pp. 425–453.

[4] D. den Hertog and C. Roos, *A survey of search directions in interior point methods for linear programming*, Math. Programming, 52 (1991), pp. 481–509.

[5] A. Fiacco and G. P. McCormick, *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*, John Wiley and Sons, New York, 1968.

[6] R. M. Freund, *Polynomial-time algorithms for linear programming based only on primal scaling and projected gradients of a potential function*, Math. Programming, 51 (1991), pp. 203–222.

[7] ———, *Projective transformations for interior point methods. Part* II: *Analysis of an algorithm for finding the weighted center of a polyhedral system*, Working paper OR180-88, Sloan School of Management, MIT, Cambridge, MA, 1988.

[8] J.-L. Goffin and J.-P. Vial, *On the computation of weighted analytic centers and dual ellipsoids with the projective algorithm*, Math. Programming, 60 (1993), pp. 93–113.

[9] D. Goldfarb and S. Liu, *An $O(n^3L)$ primal interior point algorithm for convex quadratic programming*, Math. Programming, 49 (1991), pp. 325–340.

[10] C. Gonzaga, *Search directions for interior linear programming methods*, Algorithmica, 6 (1991), pp. 153–181.

[11] ———, *Polynomial affine algorithms for linear programming*, Internal report ES-141/88, Department of Systems Engineering and Computer Sciences, COPPE-Federal University of Rio de Janeiro, Rio de Janeiro, Brazil, 1988.

[12] ———, *Large-step path-following methods for linear programming, Part* 2: *Potential reduction method*, SIAM J. Optim., 1 (1991), pp. 288–292.

[13] N. Karmarkar, *A new polynomial-time algorithm for linear programming*, Combinatorica, 4 (1984), pp. 373–395.

[14] M. Kojima, S. Mizuno, and A. Yoshishe, *A primal-dual interior point method for linear programming*, in Progress in Mathematical Programming, Interior Point and Related Methods, N. Megiddo, ed., Springer-Verlag, New York, 1989, pp. 29–47.

[15] N. Megiddo, *Pathways to the optimal set in linear programming*, in Progress in Mathematical Programming, Interior Point and Related Methods, N. Megiddo, ed., Springer-Verlag, New York, 1989, pp. 131–158.

[16] S. Mizuno, *$O(n^pL)$ iterations $O(n^3L)$ potential reduction algorithms for LP*, Linear Algebra Appl., 152 (1991), pp. 155–168.

[17] R. C. Monteiro and I. Adler, *An $O(n^3L)$ interior point algorithm. Part* I: *Linear programming*, Math. Programming, 44 (1987), pp. 27–42.

[18] J. RENEGAR, *A polynomial-time algorithm based on Newton's method for linear programming*, Math. Programming, 40 (1988), pp. 59–94.

[19] C. ROOS, *New trajectory-following polynomial-time algorithm for linear programming*, J. Optim. Theory Appl., 63 (1989), pp. 433–458.

[20] C. ROOS AND J.-P. VIAL, *Analytic centers in linear programming*, Report 88-68, Delft University of Technology, the Netherlands, 1988.

[21] ———, *A polynomial method of approximate centers for linear programming*, Math. Programming, 54 (1992), pp. 295–306.

[22] G. SONNEVEND, *An analytical centre for polyhedrons and new classes of global algorithms for linear (smooth, convex) programming*, in Lecture Notes in Control and Information Sciences 84, Springer-Verlag, New York, 1985, eds., A. Prekopa, T. Szelezson, and B. Strazicky, pp. 866–876.

[23] M. J. TODD AND B. P. BURRELL, *An extension of Karmarkar's algorithm for linear programming using dual variables*, Algorithmica, 1 (1986), pp. 409–424.

[24] M. J. TODD AND Y. YE, *A centered projective algorithm for linear programming*, Math. Oper. Res., 15 (1990), pp. 508–529.

[25] J.-P. VIAL, *A unified approach to projective algorithms for linear programming*, in Optimization, S. Dolecki, ed., Proceedings of Optimization Conference, Castel-Novel, France, 1988, Lecture Notes in Mathematics 1405, Springer-Verlag, New York, 1989, pp. 191–220.

[26] D. XIAO AND D. GOLDFARB, *A path-following projective interior point method for linear programming*, manuscript, Dept. of IE/OR, Columbia Univ., New York, 1990.

[27] H. YAMASHITA, *A polynomially and quadratically convergent method for linear programming*, manuscript, Mathematical Systems Institute Inc., Shinjuku-ku, Tokyo, 1986.

[28] Y. YE, *An $O(n^3L)$ potential reduction method for linear programming*, Math. Programming, 50 (1991), pp. 239–258.

[29] ———, *A class of projective transformations for linear programming*, SIAM J. Comput., 19 (1990), pp. 457–466.

SIAM J. OPTIMIZATION
Vol. 4, No. 1, pp. 208–227, February 1994
© 1994 Society for Industrial and Applied Mathematics
012

ON THE CONVERGENCE OF A CLASS OF INFEASIBLE INTERIOR-POINT METHODS FOR THE HORIZONTAL LINEAR COMPLEMENTARITY PROBLEM*

YIN ZHANG†

Abstract. Interior-point methods require strictly feasible points as starting points. In theory, this requirement does not seem to be particularly restrictive, but it can be costly in computation. To overcome this deficiency, most existing practical algorithms allow positive but infeasible starting points and seek feasibility and optimality simultaneously. Algorithms of this type shall be called *infeasible interior-point* algorithms. Despite their superior performance, existing infeasible interior-point algorithms still lack a satisfactory demonstration of theoretical convergence and polynomial complexity. This paper studies a popular infeasible interior-point algorithmic framework that was implemented for linear programming in the highly successful interior-point code OB1 [I. J. Lustig, R. E. Marsten, and D. F. Shanno, *Linear Algebra Appl.*, 152 (1991), pp. 191–222]. For generality, the analysis is carried out on a horizontal linear complementarity problem that includes linear and quadratic programming, as well as the standard linear complementarity problem. Under minimal assumptions, it is demonstrated that with properly controlled steps the algorithm converges at a global Q-linear rate. Moreover, with properly chosen starting points it is established the algorithm can obtain ε-feasibility and ε-complementarity in at most $O(n^2 \ln(1/\varepsilon))$ iterations.

Key words. infeasible interior-point methods, horizontal linear complementarity problem, global convergence, polynomiality

AMS subject classification. 90C05

1. Introduction.

Interior-point methods have proven to be effective for solving many optimization problems. For linear and quadratic programming, an interior point is defined as one that satisfies all constraints and, in particular, strictly satisfies all inequality constraints. Frequently, an interior point is also called a strictly feasible point. (A strictly feasible point need not always exist even when the relative interior of the feasibility set is nonempty.) As the name suggests, interior-point methods operate in the interior of a feasibility set, starting and remaining in the interior. However, finding an initial interior point is usually a nontrivial task. In theory, this difficulty can be overcome by introducing artificial variables and embedding the original feasibility set into a new one in a higher dimensional space. This embedding approach requires the use of some large but unknown parameters, as in the "big M" approach for linear programming. The disadvantages of this approach are twofold. First, it is not known a priori how large this big parameter should be and an excessively large value introduces instability into the solution process. Second, the reformulation may adversely alter the structure, such as the sparsity pattern, of the original problem due to added new columns and rows. Because of these deficiencies, embedding techniques have not been well received or widely used.

Considerable research effort has been devoted to the initialization problem of interior-point methods, especially for linear programming. A number of approaches have been proposed. Among them is the so-called combined Phase-I and Phase-II approach. In this approach, no big parameter is used, but still artificial variables

* Received by the editors June 16, 1992; accepted for publication (in revised form) December 28, 1992. This work was presented at the Fourth SIAM Conference on Optimization, Chicago, May 11–13, 1992. The research was supported in part by National Science Foundation grant DMS-9102761 and Department of Energy grant DE-FG05-91ER25100.

† Department of Mathematics and Statistics, University of Maryland Baltimore County, Baltimore, MD 21228-5398.

and constraints are introduced along with new columns and rows. Works in this direction include those by De Ghellinck and Vial [4], Anstreicher [1], [2], Todd [17], Todd and Wang [18], and Ye [20]. More recently, Kojima, Mizuno, and Yoshise [10], and Ishihara and Kojima [7] proposed methods to alleviate problems of the "big M" approach. These methods use adjustable instead of fixed big parameters. All the above works are aimed at deriving algorithms that do not require one to start from an interior point of the original problem. In these works, the primary concern was maintaining the desirable theoretical properties, such as polynomial complexity, of interior-point methods.

On the other hand, another line of research primarily aimed at practical performance has taken interior-point methods in a quite different direction. This line of research produced a series of practical algorithms along with numerical tests and comparisons, for example, see Lustig, Marsten, and Shanno [11], [12]; McShane, Monma, and Shanno [13]; and Mehrotra [15]. All these practical algorithms require neither problem reformulation nor feasible starting points. They can start from any positive but otherwise infeasible point and try to achieve feasibility and optimality simultaneously. Most of these algorithms fall into the framework of a centered and damped Newton method, as we will describe later in detail. Strictly speaking, these algorithms are no longer interior-point methods in the original sense, since the equality constraints are no longer satisfied; but most researchers continue to call them interior-point methods, partly because of the similarities in algorithmic form and partly because of the fact that the iterates generated by these algorithms are still interior points of the positive orthant, i.e., they strictly satisfy the inequality constraints $x \geq 0$.

In this paper, we will adopt the following terminology. For a given mathematical programming problem with both equality and inequality constraints, we call a point an *infeasible interior-point* for this problem if it strictly satisfies all inequality constraints but does not satisfy all equality constraints. In this terminology, the term *interior* refers to the feasibility set defined by inequality constraints only. On the other hand, a point that not only strictly satisfies all inequality constraints, but also satisfies all equality constraints is called a *feasible interior point*, or simply an *interior point*. Moreover, by an *infeasible interior-point algorithm* we mean an algorithm that generates infeasible interior-point iterates. We note that in the literature an interior point is also frequently called a *strictly feasible point*.

There are other terminologies in the literature for what we would call infeasible interior-point algorithms in this paper. In their recent work, Kojima, Megiddo, and Mizuno [8] used the term *exterior point algorithm*. On the other hand, in the classic book by Fiacco and McCormick, the term *mixed interior point-exterior point algorithm* [5, p. 59] was used for algorithms that strictly satisfy inequality constraints but do not satisfy equality constraints; while the term *exterior point algorithm* [5, p. 53] was used for a different class of algorithms.

Despite their superior practical performance, existing infeasible interior-point algorithms still lack a satisfactory demonstration of theoretical convergence, let alone a demonstration of polynomiality. Recently, Kojima, Megiddo, and Mizuno [8] analyzed a primal-dual linear programming algorithm that they called an exterior point algorithm but can be classified, by our terminology, as an infeasible interior-point algorithm. To our knowledge, their work is the first that deals with convergence of an infeasible interior-point algorithm for linear programming in the framework of the centered and damped Newton method described in §3. Their main result is that in a finite number of iterations the algorithm will produce an iterate that either satisfies

a prescribed tolerance in optimality or exceeds a prescribed upper bound in magnitude. This result, however, does not imply that, assuming the existence of a solution, the algorithm guarantees to return an approximate solution. The authors managed to obtain some slightly stronger results by modifying the standard algorithm into a nonstandard, more complicated one.

A fundamental question in the convergence analysis of infeasible interior-point methods is the following. If an algorithm were allowed to run without stopping, would the iterates converge to the solution set provided that it is nonempty? To our knowledge, this question has not yet been satisfactorily answered. Another important issue that is still open is the computational complexities of infeasible interior-point methods.

It is the objective of this paper to establish a convergence theory for an infeasible interior-point algorithmic framework that is close to the implemented practical algorithms. We consider it extremely important to analyze algorithms as close as possible to algorithms with empirically proven effectiveness. Based on this consideration, we adhere to the principle of not tailoring the algorithm just for the sake of obtaining better complexity bounds (see more discussion on this issue in the last section). As a result, we impose only mild restrictions on the choices of steplength and starting point. Our main results in this paper consist of a global Q-linear convergence result and a polynomial complexity bound.

For the sake of generality, we will study a rather general model problem, called the horizontal linear complementarity problem (horizontal LCP), that includes (but is not limited to) linear and quadratic programs, and the standard linear complementarity problem. This generality will cause little extra work in our analysis. We will describe the horizontal LCP in the next section.

We will use superscripts to denote the iteration count and subscript as indices for vector components. The norm $\|\cdot\|$ will denote the ℓ_2 norm unless otherwise specified. For $u, v \in \mathbf{R}^n$, we will also use the notation

$$\min(v) = \min_{1\le i\le n} v_i, \qquad \min(u,v) = \min_{1\le i\le n} (u_i, v_i).$$

The paper is organized as follows. In §2, we describe our model problem—a horizontal linear complementarity problem that includes linear and quadratic programs, and the standard linear complementarity problem. In §3, we give the general algorithmic framework, which is a centered and damped Newton method as implemented in OB1 [11]. In §4, we describe our choice of starting point and introduce an auxiliary sequence. This auxiliary sequence will play an important role in our analysis. In §5, we specify our choice for the steplength and state our complete algorithm. Section 6 contains technical lemmas. Our global convergence results are presented in §7. Finally, further comments are given in the last section.

2. Horizontal LCP. By the horizontal linear complementarity problem (horizontal LCP), we mean the following nonlinear system with nonnegativity constraints

$$F(x,y) = \begin{pmatrix} Mx + Ny - h \\ XYe \end{pmatrix} = 0, \qquad (x,y) \ge 0, \tag{2.1}$$

where $x, y, h, e \in \mathbf{R}^n$, $M, N \in \mathbf{R}^{n\times n}$, $X = \mathrm{diag}(x)$, $Y = \mathrm{diag}(y)$, and e has all components equal to one. The name horizontal LCP comes from the recent book by Cottle, Pang, and Stone on linear complementary problems [3]. The term *horizontal*

is used to characterize the geometric shape of the n-by-$2n$ matrix $[M\ N]$, as opposed to a vertical LCP in which the involved matrix has more rows than columns.

Problem (2.1) is sufficiently general to include the standard linear complementarity problem, linear, and quadratic programs. It is easy to see that when $N = -I$, Problem (2.1) is the standard linear complementarity problem. Moreover, consider the quadratic program

$$\begin{array}{ll} \text{minimize} & c^T x + \frac{1}{2} x^T Q x \\ \text{subject to} & Ax = b, \quad x \geq 0, \end{array} \tag{2.2}$$

where $c, x \in \mathbf{R}^n$, $b \in \mathbf{R}^m$, $A \in \mathbf{R}^{m \times n}$ has full row rank m ($m < n$), and $Q \in \mathbf{R}^{n \times n}$ is symmetric. It can be verified (see [25], for example) that the Karush–Kuhn–Tucker conditions for (2.2) can be written as (2.1) with

$$M = \begin{bmatrix} A \\ -BQ \end{bmatrix}, \quad N = \begin{bmatrix} 0 \\ B \end{bmatrix} \quad \text{and} \quad h = \begin{pmatrix} b \\ Bc \end{pmatrix}, \tag{2.3}$$

where $B \in \mathbf{R}^{(n-m) \times n}$ has full row rank $n - m$ and $BA^T = 0$, i.e., the rows of B span the null space of A. When $Q = 0$, the quadratic program (2.2) reduces to the linear program in standard form

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax = b,\ x \geq 0. \end{array} \tag{2.4}$$

We should point out that for linear or quadratic programming, the use of the B matrix in our formulation is purely for the convenience of exposition. Our analysis can be equally applied to other standard formulations. It is also worth noting that our analysis is applicable to even more general problems than the horizontal LCP. For example, one may add free variables to the horizontal LCP along with an equal number of linear equations, provided that the columns corresponding to the free variables remain linearly independent.

For convenience of reference, we define the following sets:

$$\begin{aligned} \mathcal{S} &= \{(x, y) \in \mathbf{R}^{2n} : F(x, y) = 0 \text{ and } (x, y) \geq 0\}, \\ \mathcal{A} &= \{(x, y) \in \mathbf{R}^{2n} : Mx + Ny = h\}, \\ \mathcal{F} &= \{(x, y) \in \mathcal{A} : (x, y) \geq 0\}, \\ \mathcal{F}_+ &= \{(x, y) \in \mathcal{A} : (x, y) > 0\}. \end{aligned}$$

Clearly, $\mathcal{S}$ is the solution set for the horizontal LCP (2.1). We call $\mathcal{F}$ the feasibility set of (2.1). A pair (x, y) is said to be feasible if it is in $\mathcal{F}$ and strictly feasible if it is in $\mathcal{F}_+$.

It is straightforward to show that

$$F'(x, y) = \begin{bmatrix} M & N \\ Y & X \end{bmatrix}. \tag{2.5}$$

3. Algorithmic framework. The algorithmic framework under study is a Newton-type method applied to the nonlinear system $F(x, y) = 0$ with positive starting points. At each iteration, a steplength is chosen to ensure that the next iterate remains positive, and a so-called centering step is added to the Newton step. Roughly speaking, the purpose of adding the centering step is to prevent iterates from getting too close to the boundary of the positive orthant too early. The centering step fades away as

the iterates approach a solution. As a matter of fact, the algorithmic framework is precisely the centered and damped Newton method described below.

ALGORITHM 1 (CENTERED AND DAMPED NEWTON). *Given* $(x^0, y^0) > 0$, *for* $k = 0, 1, 2, \ldots$, *do*

1. *Choose* $\sigma^k \in [0, 1)$ *and let* $\mu^k = \frac{1}{n} {x^k}^T y^k$. *Solve the following linear system for* $(\Delta x^k, \Delta y^k)$

$$F'(x^k, y^k) \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = -F(x^k, y^k) + \sigma^k \begin{pmatrix} 0 \\ \mu^k e \end{pmatrix}. \tag{3.1}$$

2. *Choose a steplength* $\alpha^k \in (0, 1]$ *and*

$$\alpha^k < \hat{\alpha}^k = \frac{-1}{\min((X^k)^{-1}\Delta x^k, (Y^k)^{-1}\Delta y^k, -1/2)}.$$

Let $x^{k+1} = x^k + \alpha^k \Delta x^k$ *and* $y^{k+1} = y^k + \alpha^k \Delta y^k$.

As one can see, a term involving $\sigma^k \mu^k e$ is added to the right-hand side of the Newton equation (3.1). This term, resulting from perturbing the equation $XYe = 0$ into $XYe = \sigma^k \mu^k e$, is commonly called the centering term. Moreover, the (centered) Newton step is damped (i.e., multiplied by $\alpha^k < 1$) whenever necessary to keep the next iterate positive. Notice that the restriction $\alpha^k < \hat{\alpha}^k$ guarantees that the iterates remain positive.

If $(x^0, y^0) \in \mathcal{F}_+$ is strictly feasible, then $\{(x^k, y^k)\} \subset \mathcal{F}_+$. In this case, Algorithm 1 is an interior-point algorithm and is called a primal-dual interior-point algorithm for the cases of linear and quadratic programming.

In order to guarantee that Algorithm 1 is well defined and to facilitate our analysis, we will make the following assumptions throughout this paper unless otherwise specified:

A1. $\mathcal{F} \neq \emptyset$, i.e., a feasible point exists.

A2. For any $(x, y) \in \mathcal{A}$ and $(\hat{x}, \hat{y}) \in \mathcal{A}$, $(x - \hat{x})^T (y - \hat{y}) \geq 0$.

It is known that Assumption A1 implies the existence of a solution to Problem (2.1) (see [6, Thm. 3.1], for example). It is also well known that Assumption A2 is satisfied by linear programs, convex quadratic programs, and monotone linear complementarity problems. Therefore, Assumptions A1–A2 are essential but by no means restrictive.

The following proposition demonstrates that Assumption A2 guarantees that Algorithm 1 is well defined. A more general version of this result can be found in [6, Thm. 2.1].

PROPOSITION 3.1. *Under Assumption A2,* $F'(x, y)$ *is nonsingular for* $(x, y) > 0$.

Proof. Let $(x, y) > 0$. Consider

$$F'(x, y) \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} Mu + Nv \\ Yu + Xv \end{pmatrix} = 0,$$

where $u, v \in \mathbf{R}^n$. Since $Mu + Nv = 0$, by Assumption A2, $u^T v \geq 0$. On the other hand, the equation $Yu + Xv = 0$ gives $u = -Y^{-1}Xv$ and $u^T v = -v^T Y^{-1} X v \leq 0$. Hence, we must have $v = 0$ and, consequently, $u = 0$. So $F'(x, y)$ is indeed nonsingular. □

For future references, we list several well-known identities as a proposition. First define

$$D^k = (Y^k)^{1/2} (X^k)^{-1/2}. \tag{3.2}$$

PROPOSITION 3.2. *Let* $\{(x^k, y^k)\}$ *be generated by Algorithm* 1. *Then*

(1) $Y^k \Delta x^k + X^k \Delta y^k = \sigma^k \mu^k e - X^k Y^k e$;

(2) $(x^k + \alpha \Delta x^k)^T (y^k + \alpha \Delta y^k) = {x^k}^T y^k (1 - \alpha + \alpha\sigma^k) + {\Delta x^k}^T \Delta y^k \alpha^2$;

(3) $\|D^k \Delta x^k\|^2 + \|(D^k)^{-1} \Delta y^k\|^2 + 2{\Delta x^k}^T \Delta y^k = \|(X^k Y^k)^{-\frac{1}{2}} (\sigma^k \mu^k e - X^k Y^k e)\|^2$.

The following proposition can also be readily verified.

PROPOSITION 3.3. *Let* $\{(x^k, y^k)\}$ *be generated by Algorithm* 1. *Then for* $k \geq 0$

$$Mx^{k+1} + Ny^{k+1} - h = (1 - \alpha^k)(Mx^k + Ny^k - h) = \nu^{k+1}(Mx^0 + Ny^0 - h),$$

where $\nu^0 = 1$ *and*

$$\nu^{k+1} = (1 - \alpha^k)\nu^k = \prod_{j=0}^{k} (1 - \alpha^j) \geq 0. \tag{3.3}$$

Moreover, if $\alpha^p = 1$ *at some iteration* p, *then* $\nu^k = 0$ *and* $(x^k, y^k) \in \mathcal{F}_+$ *for all* $k > p$.

4. Starting points and an auxiliary sequence. In this section, we specify our choice of starting point for Algorithm 1. Our starting point, strictly positive as required by Algorithm 1, is inexpensive to construct. It is not completely arbitrary, but by no means restrictive.

Given a pair $(u^0, v^0) \in \mathcal{A}$, we choose

$$(x^0, y^0) > 0 \quad \text{and} \quad (x^0, y^0) \geq (u^0, v^0). \tag{4.1}$$

For instance, one possibility is to choose $x^0 = \max(\zeta, u^0)$ and $y^0 = \max(\zeta, v^0)$ for some $\zeta > 0$. The task of finding a pair $(u^0, v^0) \in \mathcal{A}$ is straightforward and inexpensive for common applications such as linear and quadratic programming, and linear complementarity problems. Moreover, from the construction, we obviously have

$$x^0 - u^0 \geq 0, \qquad y^0 - v^0 \geq 0. \tag{4.2}$$

We will later make heavy use of an auxiliary sequence $\{(u^k, v^k)\}$, which is constructed from Algorithm 1 as follows:

$$u^{k+1} = u^k + \alpha^k(\Delta x^k + x^k - u^k) \quad v^{k+1} = v^k + \alpha^k(\Delta y^k + y^k - v^k). \tag{4.3}$$

The introduction of this auxiliary sequence was partly motivated by the two-point barrier function method recently proposed by Xu [19] for linear programming, in which two sequences are generated, one positive and another satisfying the linear constraints. In our case, however, the auxiliary sequence is used only for the purpose of analysis and need not be actually computed in Algorithm 1.

The following lemma gives useful properties of the auxiliary sequence $\{(u^k, v^k)\}$.

LEMMA 4.1. *Let* $\{(x^k, y^k)\}$ *be generated by Algorithm* 1, $\{(u^k, v^k)\}$ *be given by* (4.3), *and* $\{\nu^k\}$, *be given by* (3.3). *Then for* $k \geq 0$

(1) $(u^k, v^k) \in \mathcal{A}$;

(2) $x^{k+1} - u^{k+1} = (1 - \alpha^k)(x^k - u^k) \geq 0$ *and* $y^{k+1} - v^{k+1} = (1 - \alpha^k)(y^k - v^k) \geq 0$;

(3) $x^k - u^k = \nu^k(x^0 - u^0) \leq x^0 - u^0$ *and* $y^k - v^k = \nu^k(y^0 - v^0) \leq y^0 - v^0$;

(4) *If* $\alpha^p = 1$ *for some* $p > 0$, *then* $(x^k, y^k) = (u^k, v^k) \in \mathcal{F}_+$ *for all* $k > p$.

Proof. The proof follows from direct substitution and Proposition 3.3. □

The auxiliary sequence $\{(u^k, v^k)\}$ satisfies the first n linear equations in $F(x, y) = 0$. As the algorithm progresses, the distance between $\{(u^k, v^k)\}$ and $\{(x^k, y^k)\}$ decreases monotonically. We hope that feasibility will be attained as the distance becomes zero. The fact that $x^k - u^k$ and $y^k - v^k$ are always nonnegative will prove particularly useful in §6.

5. Steplength and the complete algorithm. In this section, we will use the following notation:

$$x(\alpha) = x^k + \alpha \Delta x^k, \qquad y(\alpha) = y^k + \alpha \Delta y^k.$$

Moreover, when $\alpha = 0$, we have $x^k = x(0)$, $y^k = y(0)$, and so on; and when $\alpha = \alpha^k$, we have $x^{k+1} = x(\alpha^k)$, $y^{k+1} = y(\alpha^k)$, and so on.

There are two control parameters in Algorithm 1: the centering parameter σ^k and the steplength α^k. In practical implementations, σ^k is often set to a small constant, and α^k is chosen to be a large fraction, usually 99% or more, of $\hat{\alpha}^k$—the length of the step to the boundary of the positive orthant. This choice for α^k may be adequate for practical purposes, but we need more elaborated choices for theoretically guaranteed convergence. In this section, we specify our choice for α^k. Our choice is based on several factors, including decreasing a merit function that is described below.

We are seeking two objectives: complementarity and feasibility. The most commonly used quantity for measuring complementarity is $x^T y$. For feasibility, it is natural to use the residual norm $\|Mx + Ny - h\|$ as a measure. Therefore, we define our merit function as

$$\phi(x, y) = x^T y + \|Mx + Ny - h\|. \tag{5.1}$$

The values of the merit function at the iterates will be denoted by

$$\phi^k = \phi(x^k, y^k).$$

For notational convenience, we define

$$r(\alpha) = \|Mx(\alpha) + Ny(\alpha) - h\| \quad \text{and} \quad \phi(\alpha) = x(\alpha)^T y(\alpha) + r(\alpha).$$

By our notational convention, we have $r^k = r(0)$ and $r^{k+1} = r(\alpha^k)$. In addition, Proposition 3.3 and Lemma 4.1 imply

$$r^{k+1} = (1 - \alpha^k) r^k = \nu^{k+1} r^0. \tag{5.2}$$

Moreover, letting $\nu(\alpha) = (1 - \alpha)\nu^k$ and using Proposition 3.2 (2) and (5.2), we have

$$\phi(\alpha) = x(\alpha)^T y(\alpha) + \nu(\alpha) r^0 = (1 - \delta(\alpha))\phi^k,$$

where

$$\delta(\alpha) = \frac{\alpha[(1 - \sigma^k) x^{k^T} y^k + \nu^k r^0 - \Delta x^{k^T} \Delta y^k \alpha]}{x^{k^T} y^k + \nu^k r^0}. \tag{5.3}$$

Observe from (5.3) that it is possible to choose $\alpha^k \in (0, 1]$ such that

$$0 < \delta^k = \delta(\alpha^k) < 1.$$

Consequently, the sequence $\{\phi^k\}$ generated by Algorithm 1 satisfies

$$\phi^{k+1} = (1 - \delta^k)\phi^k < \phi^k. \tag{5.4}$$

Furthermore, observe that $\{\phi^k\}$ converges to zero Q-linearly if $\delta^k \geq \delta > 0$.

Our choice of the steplength α^k is based on several considerations. Since our goal is to drive the merit function to zero, we would like to maximize $\delta(\alpha)$ in $(0, 1]$ at each

iteration. Meanwhile, to obtain global convergence, we need to enforce two conditions specified below. Let us first define a constant $\gamma \in (0,1)$ that satisfies

$$\gamma \leq \min(X^0 Y^0 e)/({x^0}^T y^0/n). \tag{5.5}$$

Without loss of generality, we can assume that γ is independent of n. The two conditions on the choice of α^k are that for all $\alpha \in (0, \alpha^k] \subset (0,1]$,

$$\min(X(\alpha)Y(\alpha)e) \geq \gamma x(\alpha)^T y(\alpha)/n, \tag{5.6}$$

and

$$x(\alpha)^T y(\alpha) \geq \nu(\alpha){x^0}^T y^0 = (1-\alpha)\nu^k {x^0}^T y^0. \tag{5.7}$$

From the continuity of $(x(\alpha), y(\alpha))$, it should be clear that if α^k satisfies (5.6) and (5.7), then

$${x^{k+1}}^T y^{k+1} = x(\alpha^k)^T y(\alpha^k) \geq 0 \quad \text{and} \quad (x^{k+1}, y^{k+1}) \geq 0.$$

Condition (5.6) is essential for interior-point methods. Its role is to prevent iterates from prematurely getting too close to the boundary of the positive orthant. Compared with many interior-point algorithms, our restriction (5.6) on γ is quite liberal. For example, γ can be chosen to be very small as long as it is independent of n. This means that our requirement of centrality for the iterates is very mild.

In addition, observe that condition (5.7) implies that

$$\frac{{x^k}^T y^k}{{x^0}^T y^0} \geq \nu^k = \frac{r^k}{r^0}.$$

So condition (5.7) gives a priority to feasibility over complementarity. Roughly speaking, we do not want to have complementarity without feasibility.

It is now widely accepted that in order for interior-point algorithms to obtain good numerical performance, large steplengths (eventually close to one or approaching one) should be taken. We believe that conditions (5.6) and (5.7) do not prevent an algorithm from eventually taking large steps; though this issue deserves further study.

Observe that ${x^p}^T y^p = 0$ is possible only if $\nu^p = 0$. In this case of ${x^p}^T y^p = 0$, (x^p, y^p) will be a solution to (2.1) and Algorithm 1 can be terminated. However, it seems extremely unlikely for ${x^p}^T y^p = 0$ to happen in practice. Without loss of generality, we will not consider this finite termination case but assume that $\{(x^k, y^k)\}$ is an infinite sequence.

For notational convenience, we rewrite conditions (5.6) and (5.7) as follows. For all $\alpha \in (0, \alpha^k] \subset (0,1]$

$$g_1(\alpha) = \min(X(\alpha)Y(\alpha)e) - \gamma x(\alpha)^T y(\alpha)/n \geq 0 \tag{5.8}$$

and

$$g_2(\alpha) = x(\alpha)^T y(\alpha) - \nu(\alpha){x^0}^T y^0 \geq 0. \tag{5.9}$$

It is clear that $g_1(\alpha)$ is a piecewise quadratic and $g_2(\alpha)$ is a quadratic.

At each iteration, our objective is to minimize the merit function subject to both (5.8) and (5.9). Hence, we choose α^k as

$$\alpha^k = \arg \max_{\alpha \in (0,1]} \{\delta(\alpha) : g_i(\beta) \geq 0 \text{ for all } \beta \leq \alpha,\ i = 1,2\}. \tag{5.10}$$

Since $\delta(\alpha)$ is a quadratic, it is not difficult to determine the solution of (5.10). The lemma below gives the solution.

LEMMA 5.1. *If conditions* (5.6) *and* (5.7) *are satisfied by* $\alpha = 0$, *then problem* (5.10) *has the unique solution*

$$\alpha^k = \begin{cases} \min(1, \alpha_1^k, \alpha_2^k), & \text{if } \Delta x^{k^T} \Delta y^k \le 0, \\ \min(1, \alpha_1^k, \alpha_3^k), & \text{if } \Delta x^{k^T} \Delta y^k > 0, \end{cases} \tag{5.11}$$

where

$$\alpha_1^k = \min\{\alpha > 0 : g_1(\alpha) = 0\}, \tag{5.12}$$

$$\alpha_2^k = \begin{cases} 1 & \text{if } \nu^k = 0 \text{ or } \Delta x^{k^T} \Delta y^k \ge 0, \\ \min\{\alpha > 0 : g_2(\alpha) = 0\} & \text{otherwise,} \end{cases} \tag{5.13}$$

and

$$\alpha_3^k = \frac{(1-\sigma^k) x^{k^T} y^k + \nu^k r^0}{2 \Delta x^{k^T} \Delta y^k}. \tag{5.14}$$

Proof. We first verify the existence of α_1^k and α_2^k. The existence of $\alpha_1^k > 0$ is known, see [22] for a proof. If $\nu^k = 0$, then (5.6) implies (5.7), i.e., $g_2(\alpha) \ge 0$ holds for all $\alpha \in (0, \alpha_1^k]$. Moreover, when $\Delta x^{k^T} \Delta y^k \ge 0$,

$$g_2(\alpha) = (1-\alpha)(x^{k^T} y^k - \nu^k x^{0^T} y^0) + \alpha(\sigma^k x^{k^T} y^k + \alpha \Delta x^{k^T} \Delta y^k) \ge 0$$

holds for all $\alpha \in (0, 1]$. This means that in the case of $\nu^k = 0$ or $\Delta x^{k^T} \Delta y^k \ge 0$, (5.9) becomes redundant, leading to the first relation in (5.13) without loss of generality. Otherwise, a positive and finite-valued α_2^k is defined by the second relation in (5.13).

When $\Delta x^{k^T} \Delta y^k \le 0$, the maximum of the quadratic $\delta(\alpha)$ is reached at the boundary of the feasible interval, confirming the first relation in (5.11). When $\Delta x^{k^T} \Delta y^k > 0$, it is easy to see that α_3^k is the global maximizer of the quadratic $\delta(\alpha)$. So $\delta(\alpha)$ attains its constrained maximum at α_3^k whenever it is inside the feasible interval $(0, \min(1, \alpha_1^k))$ (note that $\alpha_2^k = 1$ in this case); otherwise the constrained maximum will be attained at the boundary. This confirms the second relation in (5.11). It also proves the lemma. □

As mentioned earlier, $g_1(\alpha)$ is a piecewise quadratic and $g_2(\alpha)$ is a quadratic. Finding the roots of $g_1(\alpha)$ requires at most $O(n)$ operations.

Now we are in a position to state a complete infeasible interior-point algorithm for solving the horizontal LCP.

ALGORITHM 2 (INFEASIBLE INTERIOR-POINT ALGORITHM). *Let* (x^0, y^0) *satisfy* (4.1). *For* $k = 0, 1, 2, \ldots$, *do the following*:

1. *Choose* $\sigma^k \in (0, 1)$ *and let* $\mu^k = \frac{1}{n} x^{k^T} y^k$. *Solve the linear system* (3.1) *for* $(\Delta x^k, \Delta y^k)$.
2. *Set the steplength* α^k *by* (5.11). *Let* $x^{k+1} = x^k + \alpha^k \Delta x^k$ *and* $y^{k+1} = y^k + \alpha^k \Delta y^k$.

Obviously, Algorithm 2 is a special case of Algorithm 1. So all the properties of Algorithm 1 previously stated also apply to Algorithm 2.

6. Technical lemmas. In this section, we prove several more technical lemmas. These technical results will be used to establish our main convergence results.

LEMMA 6.1. *Let* $\{(x^k, y^k)\}$ *be generated by Algorithm* 2. *For any* $(x^*, y^*) \in \mathcal{S}$,

$$\frac{(x^k - u^k)^T y^k + (y^k - v^k)^T x^k}{x^{k^T} y^k} \leq 1 + \frac{(x^0 - u^0)^T y^* + (y^0 - v^0)^T x^* + (x^0 - u^0)^T (y^0 - v^0)}{x^{0^T} y^0}.$$

Moreover, if $\mathcal{F}_+ \neq \emptyset$, *then* $\{(x^k, y^k)\}$ *is bounded.*

Proof. Let $(\hat{x}, \hat{y}) \in \mathcal{F}$. Consider

$$\begin{aligned}
&\hat{x}^T \hat{y} + x^{k^T} y^k - (\hat{x}^T y^k + \hat{y}^T x^k) \\
&\quad = (\hat{x} - x^k)^T (\hat{y} - y^k) \\
&\quad = [(\hat{x} - u^k) - (x^k - u^k)]^T [(\hat{y} - v^k) - (y^k - v^k)] \\
&\quad = (\hat{x} - u^k)^T (\hat{y} - v^k) - (x^k - u^k)^T (\hat{y} - v^k) \\
&\qquad - (\hat{x} - u^k)^T (y^k - v^k) + (x^k - u^k)^T (y^k - v^k).
\end{aligned}$$

In view of Assumption A2 and Lemma 4.1(1),

$$(\hat{x} - u^k)^T (\hat{y} - v^k) \geq 0.$$

Hence,

$$\begin{aligned}
\hat{x}^T y^k + \hat{y}^T x^k \leq \hat{x}^T \hat{y} + x^{k^T} y^k + (x^k - u^k)^T (\hat{y} - v^k) \\
+ (y^k - v^k)^T (\hat{x} - u^k) - (x^k - u^k)^T (y^k - v^k).
\end{aligned} \tag{6.1}$$

Replacing $\hat{x} - u^k$ by $(\hat{x} - x^k) + (x^k - u^k)$ and $\hat{y} - v^k$ by $(\hat{y} - y^k) + (y^k - v^k)$ in (6.1), we can rewrite the last three terms in (6.1) into

$$(x^k - u^k)^T (\hat{y} - y^k) + (y^k - v^k)^T (\hat{x} - x^k) + (x^k - u^k)^T (y^k - v^k).$$

Substituting the above into (6.1) and rearranging, we obtain

$$\begin{aligned}
&\hat{x}^T y^k + \hat{y}^T x^k + (x^k - u^k)^T y^k + (y^k - v^k)^T x^k \\
&\quad \leq \hat{x}^T \hat{y} + x^{k^T} y^k + (x^k - u^k)^T \hat{y} + (y^k - v^k)^T \hat{x} + (x^k - u^k)^T (y^k - v^k).
\end{aligned} \tag{6.2}$$

Note that $x^{k^T} y^k \leq \phi^k \leq \phi^0$, $(x^k, y^k) > 0$ and (see Lemma 4.1)

$$0 \leq x^k - u^k \leq x^0 - u^0 \quad \text{and} \quad 0 \leq y^k - v^k \leq y^0 - v^0.$$

It follows from (6.2) and (5.7) that for any $(x^*, y^*) \in \mathcal{S}$

$$\begin{aligned}
&[(x^k - u^k)^T y^k + (y^k - v^k)^T x^k] / x^{k^T} y^k \\
&\quad \leq 1 + [(x^k - u^k)^T y^* + (y^k - v^k)^T x^* + (x^k - u^k)^T (y^k - v^k)] / x^{k^T} y^k \\
&\quad = 1 + \nu^k [(x^0 - u^0)^T y^* + (y^0 - v^0)^T x^* + (x^0 - u^0)^T (y^k - v^k)] / x^{k^T} y^k \\
&\quad \leq 1 + [(x^0 - u^0)^T y^* + (y^0 - v^0)^T x^* + (x^0 - u^0)^T (y^k - v^k)] / x^{0^T} y^0 \\
&\quad \leq 1 + [(x^0 - u^0)^T y^* + (y^0 - v^0)^T x^* + (x^0 - u^0)^T (y^0 - v^0)] / x^{0^T} y^0.
\end{aligned}$$

This proves the first part of the lemma. Moreover, if $\mathcal{F}_+ \neq \emptyset$, then it follows from (6.2) that for any fixed $(\hat{x}, \hat{y}) \in \mathcal{F}_+$

$$\hat{x}^T y^k + \hat{y}^T x^k \leq \hat{x}^T \hat{y} + \phi^0 + (x^0 - u^0)^T \hat{y} + (y^0 - v^0)^T \hat{x} + (x^0 - u^0)^T (y^0 - v^0).$$

This proves the boundedness of $\{(x^k, y^k)\}$ and completes the proof. □

Before we proceed to the next lemma, let us define the following quantities:

$$\xi^k = \left(\frac{n}{\gamma}\right)^{1/2} \frac{(x^k - u^k)^T y^k + (y^k - v^k)^T x^k}{x^{k^T} y^k}, \tag{6.3}$$

$$\eta^k = (1 - 2\sigma^k + (\sigma^k)^2/\gamma) + 2\nu^k (x^0 - u^0)^T (y^0 - v^0)/x^{0^T} y^0. \tag{6.4}$$

and

$$\omega^k = \left(\xi^k + \sqrt{(\xi^k)^2 + \eta^k}\right)^2. \tag{6.5}$$

LEMMA 6.2. *Let* $\{(x^k, y^k)\}$ *and* $\{(\Delta x^k, \Delta y^k)\}$ *be generated by Algorithm* 2, *and* D^k *be given by* (3.2). *Then*

$$\|D^k \Delta x^k\|^2 + \|(D^k)^{-1} \Delta y^k\|^2 \leq \omega^k x^{k^T} y^k, \tag{6.6}$$

Moreover, the sequence $\{\omega^k\}$ *is bounded.*

Proof. Let

$$t^k = (\|D^k \Delta x^k\|^2 + \|(D^k)^{-1} \Delta y^k\|^2)^{1/2}. \tag{6.7}$$

Notice that

$$\|D^k \Delta x^k\| \leq t^k \quad \text{and} \quad \|(D^k)^{-1} \Delta y^k\| \leq t^k. \tag{6.8}$$

Since $(u^k, v^k) \in \mathcal{A}$ by Lemma 4.1, it can be easily verified that

$$M(\Delta x^k + x^k - u^k) + N(\Delta y^k + y^k - v^k) = 0.$$

Hence, it follows from Assumption A2 that

$$(\Delta x^k + x^k - u^k)^T (\Delta y^k + y^k - v^k) \geq 0.$$

Using this fact and (6.8), we have the following estimate

$$\begin{aligned}
\Delta x^{k^T} \Delta y^k &= [(\Delta x^k + x^k - u^k) - (x^k - u^k)]^T [(\Delta y^k + y^k - v^k) - (y^k - v^k)] \\
&= (\Delta x^k + x^k - u^k)^T (\Delta y^k + y^k - v^k) - (\Delta x^k + x^k - u^k)^T (y^k - v^k) \\
&\quad - (x^k - u^k)^T (\Delta y^k + y^k - v^k) + (x^k - u^k)^T (y^k - v^k) \\
&\geq -(x^k - u^k)^T \Delta y^k - (y^k - v^k)^T \Delta x^k - (x^k - u^k)^T (y^k - v^k) \\
&= -[D^k (x^k - u^k)]^T [(D^k)^{-1} \Delta y^k] - [(D^k)^{-1} (y^k - v^k)]^T [D^k \Delta x^k] \\
&\quad - (x^k - u^k)^T (y^k - v^k) \\
&\geq -e^T D^k (x^k - u^k) t^k - e^T (D^k)^{-1} (y^k - v^k) t^k - (x^k - u^k)^T (y^k - v^k).
\end{aligned}$$

Therefore,

$$\begin{aligned}\Delta x^{k^T}\Delta y^k \geq &-[e^T D^k(x^k-u^k)+e^T(D^k)^{-1}(y^k-v^k)]t^k\\ &-(x^k-u^k)^T(y^k-v^k).\end{aligned} \tag{6.9}$$

Let us further examine the two terms involving D^k in the above inequality. It follows from condition (5.6) that

$$\frac{x^{k^T}y^k}{x_i^k y_i^k} \leq \frac{n}{\gamma}.$$

This leads to

$$(x^{k^T}y^k)^{1/2}D^k = [x^{k^T}y^k(X^kY^k)^{-1}]^{1/2}Y^k \leq \left(\frac{n}{\gamma}\right)^{1/2} Y^k,$$

and

$$\begin{aligned}e^T D^k(x^k-u^k) &= (x^{k^T}y^k)^{1/2}e^T[(x^{k^T}y^k)^{1/2}D^k](x^k-u^k)/x^{k^T}y^k\\ &\leq (nx^{k^T}y^k/\gamma)^{1/2}e^T Y^k(x^k-u^k)/x^{k^T}y^k\\ &= (nx^{k^T}y^k/\gamma)^{1/2}(x^k-u^k)^T y^k/x^{k^T}y^k.\end{aligned}$$

Similarly,

$$e^T(D^k)^{-1}(y^k-v^k) \leq \left(\frac{nx^{k^T}y^k}{\gamma}\right)^{1/2} \frac{(y^k-v^k)^T x^k}{x^{k^T}y^k}.$$

Thus, from (6.9) and the above two inequalities we have

$$\Delta x^{k^T}\Delta y^k \geq -\left(\frac{nx^{k^T}y^k}{\gamma}\right)^{1/2} \frac{(x^k-u^k)^T y^k+(y^k-v^k)^T x^k}{x^{k^T}y^k}t^k - (x^k-u^k)^T(y^k-v^k),$$

or equivalently,

$$\Delta x^{k^T}\Delta y^k \geq -(x^{k^T}y^k)^{1/2}\xi^k t^k - (x^k-u^k)^T(y^k-v^k). \tag{6.10}$$

Furthermore, from Lemma 4.1 and condition (5.7)

$$\begin{aligned}(x^k-u^k)^T(y^k-v^k) &\leq \nu^k(x^0-u^0)^T(y^k-v^k)\\ &\leq x^{k^T}y^k(x^0-u^0)^T(y^0-v^0)/x^{0^T}y^0.\end{aligned} \tag{6.11}$$

In view of Proposition 3.2(3), (6.10), and (6.11)

$$\begin{aligned}&x^{k^T}y^k[1-2\sigma^k+(\sigma^k)^2/\gamma]\\ &\quad\geq \|(X^kY^k)^{-1/2}(\sigma^k\mu^k e - X^kY^k e)\|^2 = (t^k)^2 + 2\Delta x^{k^T}\Delta y^k\\ &\quad\geq (t^k)^2 - 2(x^{k^T}y^k)^{1/2}\xi^k t^k - 2(x^k-u^k)^T(y^k-v^k)\\ &\quad\geq (t^k)^2 - 2(x^{k^T}y^k)^{1/2}\xi^k t^k - 2\nu^k x^{k^T}y^k(x^0-u^0)^T(y^0-v^0)/x^{0^T}y^0.\end{aligned}$$

This leads to the following inequality

$$(t^k)^2 - 2(x^{k^T}y^k)^{1/2}\xi^k t^k - x^{k^T}y^k\eta^k \leq 0.$$

The quadratic $t^2 - 2({x^k}^T y^k)^{1/2}\xi^k t - {x^k}^T y^k \eta^k$ is convex and has a unique positive root at

$$t = \left(\xi^k + \sqrt{(\xi^k)^2 + \eta^k}\right)({x^k}^T y^k)^{1/2}.$$

This implies

$$(t^k)^2 \le \left(\xi^k + \sqrt{(\xi^k)^2 + \eta^k}\right)^2 {x^k}^T y^k = \omega^k {x^k}^T y^k \tag{6.12}$$

and verifies (6.6).

Finally, it is evident that $\{\eta^k\}$ is bounded. And so is $\{\xi^k\}$ by Lemma 6.1. Consequently, $\{\omega^k\}$ is bounded. This completes the proof. □

As a result of the boundedness of $\{\omega^k\}$, we have

$$(t^k)^2 = \|D^k \Delta x^k\|^2 + \|(D^k)^{-1}\Delta y^k\|^2 \le \omega {x^k}^T y^k, \tag{6.13}$$

where

$$\omega = \lim_{k\to\infty} \sup \omega^k < +\infty. \tag{6.14}$$

It is worth noting that when (x^k, y^k) is feasible, we have $x^k - u^k = y^k - v^k = 0$ and $\nu^k = 0$. This implies that

$$\xi^k = 0 \quad \text{and} \quad \eta^k = 1 - 2\sigma^k + (\sigma^k)^2/\gamma,$$

and in turn

$$\|D^k \Delta x^k\|^2 + \|(D^k)^{-1}\Delta y^k\|^2 \le (1 - 2\sigma^k + (\sigma^k)^2/\gamma){x^k}^T y^k.$$

The above is a well-known bound for interior-point methods directly following from Proposition 3.2(3). In view of this, it appears that (6.13) is a reasonable extension of the above inequality from the feasible interior-point framework to the infeasible interior-point framework.

The next lemma gives lower bounds to α^k and σ^k. We will use the following inequality:

$$|\Delta {x^k}^T \Delta y^k| \le \|D^k \Delta x^k\| \|(D^k)^{-1}\Delta y^k\| \le \frac{(t^k)^2}{2}, \tag{6.15}$$

where t^k is defined by (6.7) and the second inequality follows from the geometric-arithmetic inequality.

LEMMA 6.3. *Let α^k be given by Lemma 5.1 and δ^k by (5.3). Then*

(1) $\alpha^k \ge \min\{1, (1-\gamma)\hat{\sigma}^k({x^k}^T y^k/n)/(t^k)^2\}$, *where* $\hat{\sigma}^k = \min(\sigma^k, 1-\sigma^k)$;

(2) $\delta^k \ge \alpha^k(1 - \sigma^k - \alpha^k|\Delta {x^k}^T \Delta y^k|/{x^k}^T y^k)$.

Proof. (1) It suffices to show that for $i = 1, 2, 3$,

$$\alpha_i^k \ge \min\left(1, \frac{(1-\gamma)\hat{\sigma}^k {x^k}^T y^k/n}{(t^k)^2}\right). \tag{6.16}$$

For $i = 1$, it follows from a similar proof to that of Lemma 3.4 in [22] that α_i^k satisfies

$$\alpha_1^k \geq \min\left(1, \frac{(1-\gamma)\sigma^k {x^k}^T y^k/n}{2\|\Delta x^k \Delta y^k\|_\infty}\right),$$

where $\Delta X^k = \mathrm{diag}(\Delta x^k)$. It implies (6.16) because

$$\|\Delta X^k \Delta y^k\|_\infty \leq \|D^k \Delta x^k\| \|(D^k)^{-1} \Delta y^k\| \leq \frac{(t^k)^2}{2}.$$

For $i = 2$, we need only to consider the case of $\nu^k > 0$ and ${\Delta x^k}^T \Delta y^k < 0$. In this case, it can be verified by substitution that

$$g_2(\alpha) = (1-\alpha)({x^k}^T y^k - \nu^k {x^0}^T y^0) + \alpha(\sigma^k {x^k}^T y^k + \alpha {\Delta x^k}^T \Delta y^k).$$

Since ${x^k}^T y^k - \nu^k {x^0}^T y^0 \geq 0$, it is clear that $g_2(\alpha) \geq 0$ at

$$\alpha = \min(1, -\sigma^k {x^k}^T y^k / {\Delta x^k}^T \Delta y^k).$$

By (5.13) and (6.15)

$$\alpha_2^k \geq \min\left(1, -\frac{\sigma^k {x^k}^T y^k}{{\Delta x^k}^T \Delta y^k}\right) \geq \min\left(1, \frac{2\sigma^k {x^k}^T y^k}{(t^k)^2}\right).$$

So α_2^k also satisfies (6.16).

For $i = 3$, from (5.14), α_3^k satisfies

$$\alpha_3^k \geq \frac{(1-\sigma^k){x^k}^T y^k}{2{\Delta x^k}^T \Delta y^k} \geq \frac{\hat{\sigma}^k {x^k}^T y^k}{(t^k)^2}$$

and thus (6.16). This proves (1).

(2) From (5.3),

$$\begin{aligned}
\delta(\alpha) &= \alpha[(1-\sigma^k){x^k}^T y^k + \nu^k r^0 - {\Delta x^k}^T \Delta y^k \alpha]/({x^k}^T y^k + \nu^k r^0) \\
&= \alpha[1 - \sigma^k + (\sigma^k \nu^k r^0 - {\Delta x^k}^T \Delta y^k \alpha)/({x^k}^T y^k + \nu^k r^0)] \\
&\geq \alpha[1 - \sigma^k - \alpha {\Delta x^k}^T \Delta y^k/({x^k}^T y^k + \nu^k r^0)] \\
&\geq \alpha[1 - \sigma^k - \alpha|{\Delta x^k}^T \Delta y^k|/({x^k}^T y^k + \nu^k r^0)] \\
&\geq \alpha(1 - \sigma^k - \alpha|{\Delta x^k}^T \Delta y^k|/{x^k}^T y^k).
\end{aligned}$$

This completes the proof. □

7. Global convergence. In this section, we present our main convergence results for Algorithm 2 under Assumptions A1–A2. We first give a range for the choice of the centering parameter σ^k:

$$0 < \sigma \leq \sigma^k \leq \tfrac{1}{2}. \tag{7.1}$$

We should point out that the upper bound $\frac{1}{2}$ is chosen for simplicity. It could be any number in $(0, 1)$ without affecting our results. Our first result is a global Q-linear convergence result.

THEOREM 7.1. *Let $\{\phi^k\}$ be generated by Algorithm 2 with σ^k satisfying (7.1). Then $\{\phi^k\}$ converges to zero at a global Q-linear rate, i.e., there exists $\delta \in (0,1)$ such that*

$$\phi^{k+1} \le (1-\delta)\phi^k, \qquad k = 0, 1, 2, \ldots.$$

Proof. In view of (5.4), we need only show that $\delta^k \ge \delta > 0$ for some $\delta \in (0,1)$. By Lemma 6.3(2), it will suffice to show that α^k is bounded away from zero and $|\Delta x^{k^T} \Delta y^k| / x^{k^T} y^k$ is bounded above.

Without loss of generality, assume $\omega \ge 1$. Substituting (6.13) into Lemma 6.3(1) and noting that (7.1) implies $\hat{\sigma}^k = \sigma^k \ge \sigma$, we obtain

$$\alpha^k \ge \min\left(1, \frac{(1-\gamma)\sigma}{n\omega}\right) \ge \frac{(1-\gamma)\sigma}{n\omega}. \tag{7.2}$$

In view of (6.15) and Lemma 6.2,

$$\frac{|\Delta x^{k^T} \Delta y^k|}{x^{k^T} y^k} \le \frac{(t^k)^2}{2 x^{k^T} y^k} \le \frac{\omega}{2}.$$

Recall that δ^k is the maximum of $\delta(\alpha)$ in $(0, \alpha^k]$. Therefore, for $\alpha \in (0, \alpha^k]$

$$\delta^k \ge \left(1 - \sigma^k - \frac{\omega}{2}\alpha\right)\alpha \ge \frac{(1-\omega\alpha)\alpha}{2}.$$

Let α be the right-hand side of (7.2), then

$$\delta^k \ge \left(1 - \frac{(1-\gamma)\sigma}{n}\right)\frac{(1-\gamma)\sigma}{2n\omega} > 0. \tag{7.3}$$

This completes the proof. □

We observe that the key quantity involved in the estimates of δ^k is ω, and in turn the key quantity involved in ω is the magnitude of infeasibility. Therefore, from our analysis the initial convergence rate of Algorithm 2 depends on the initial infeasibility. Under further specifications on the starting point, we will establish a polynomial bound within which Algorithm 2 can produce an approximate solution with ε-feasibility and ε-complementarity.

Let us consider specific choices for $(u^0, v^0) \in \mathcal{A}$ and $(x^0, y^0) > 0$. Since the matrix $[M\ N]$ has full row rank (see Proposition 3.1), we can define

$$\begin{pmatrix} u^0 \\ v^0 \end{pmatrix} = \begin{bmatrix} M^T \\ N^T \end{bmatrix} [MM^T + NN^T]^{-1} h. \tag{7.4}$$

Since (u^0, v^0) is the minimum-norm solution of the linear system $Mx + Ny = h$, we have

$$\|(u^0, v^0)\| = \min\{\|(x,y)\| : (x,y) \in \mathcal{A}\} \le \min\{\|(x^*, y^*)\| : (x^*, y^*) \in \mathcal{S}\} = \rho^*. \tag{7.5}$$

One possible choice for (x^0, y^0) is as follows:

$$(x^0, y^0) = \rho(e, e), \qquad \rho \ge \|(u^0, v^0)\|. \tag{7.6}$$

Clearly, $(x^0, y^0) > 0$ and $(x^0, y^0) \ge (u^0, v^0)$, i.e., (x^0, y^0) satisfies (4.1). Moreover, we have the following lemma.

LEMMA 7.2. *Let* (x^0, y^0) *and* (u^0, v^0) *be defined by* (7.6) *and* (7.4), *respectively, and* $\{\omega^k\}$ *by* (6.14). *If*

$$\rho \geq \rho^*/(\beta\sqrt{n}) \tag{7.7}$$

for some positive constant β *independent of* n, *then*

$$\omega = \lim_{k\to\infty} \sup \omega^k = O(n).$$

Proof. We first show that there exist positive constants β_1 and β_2, both independent of n, such that for some $(x^*, y^*) \in \mathcal{S}$

$$[(x^0 - u^0)^T y^* + (y^0 - v^0)^T x^*]/x^{0^T} y^0 \leq \beta_1, \tag{7.8}$$

and

$$(x^0 - u^0)^T (y^0 - v^0)/x^{0^T} y^0 \leq \beta_2. \tag{7.9}$$

From the definition of ρ^* in (7.5), we can choose $(x^*, y^*) \in \mathcal{S}$ such that

$$\|x^*\| \leq \rho^* \quad \text{and} \quad \|y^*\| \leq \rho^*. \tag{7.10}$$

It follows from the construction of (u^0, v^0) and (x^0, y^0) that

$$\|x^0 - u^0\| \leq \|x^0\| + \|u^0\| \leq \rho(\sqrt{n} + 1), \tag{7.11}$$

and

$$\|y^0 - v^0\| \leq \|y^0\| + \|v^0\| \leq \rho(\sqrt{n} + 1). \tag{7.12}$$

Moreover, (7.6) and (7.7) together imply

$$x^{0^T} y^0 = \rho^2 n \geq \rho\rho^* \sqrt{n}/\beta. \tag{7.13}$$

It follows from the Cauchy–Schwarz inequality, (7.10), and (7.11) that

$$(x^0 - u^0)^T y^* \leq \|x^0 - u^0\| \|y^*\| \leq \rho\rho^*(\sqrt{n} + 1).$$

Similarly, we can show that

$$(y^0 - v^0)^T x^* \leq \rho\rho^*(\sqrt{n} + 1),$$

and also

$$(x^0 - u^0)^T (y^0 - v^0) \leq \rho^2(\sqrt{n} + 1)^2.$$

Hence, in view of (7.13) we have that (7.8) holds for

$$\beta_1 = 4\beta \geq 2\beta \frac{\sqrt{n} + 1}{\sqrt{n}};$$

and that (7.9) holds for

$$\beta_2 = 4 \geq \frac{(\sqrt{n} + 1)^2}{n}.$$

Considering Lemma 6.1 and substituting (7.8) and (7.9) into (6.3) and (6.4), we see that

$$\xi^k \le \left(\frac{n}{\gamma}\right)^{1/2}(1+\beta_1+\beta_2) = \left(\frac{n}{\gamma}\right)^{1/2}(4\beta+5),$$

and

$$\eta^k \le 1 - 2\sigma^k + (\sigma^k)^2/\gamma + 2\beta_2 \le 1/\gamma + 9.$$

Finally, substituting the above bounds for ξ^k and η^k into (6.5), the definition of ω^k, and taking the upper limit, we have $\omega = O(n)$. This completes the proof. □

Using Theorem 7.1 and Lemma 7.2, we obtain the following complexity result for Algorithm 2.

THEOREM 7.3. *Let* (x^0, y^0) *be given by* (7.6) *with* ρ *satisfying* (7.7), *and let* $\{\phi^k\}$ *be generated by Algorithm* 2 *with* σ^k *satisfying* (7.1). *Assume that for given* $\varepsilon > 0$

$$\phi^0 = {x^0}^T y^0 + r^0 \le 1/\varepsilon^\tau, \tag{7.14}$$

where $\tau > 0$ *is independent of* n. *There exists a positive constant*

$$K_\varepsilon = O(n^2 \ln(1/\varepsilon)) \tag{7.15}$$

such that for $k \ge K_\varepsilon$

$$\phi^k \le \varepsilon.$$

Proof. It follows from Theorem 7.1 and a standard argument that $\phi^k \le \varepsilon$ for

$$k \ge \frac{\ln(\varepsilon/\phi^0)}{\ln(1-\delta)} \ge \frac{O(\ln(1/\varepsilon))}{\delta} = K_\varepsilon,$$

where we may take $\delta = \liminf \delta^k$. From (7.3)

$$\delta \ge \left(1 - \frac{2(1-\gamma)\sigma}{n}\right)\frac{(1-\gamma)\sigma}{n\omega} \quad \text{or} \quad \frac{1}{\delta} = O(n\omega).$$

Hence,

$$K_\varepsilon = O(n\omega \ln(1/\varepsilon)).$$

By Lemma 7.2, we obtain (7.15) and this completes the proof. □

8. Stopping criteria. We note that Algorithm 2 is a generic algorithm that does not contain stopping criteria. Theorem 7.1 states that as long as there exists a feasible point, Algorithm 2 always converges in the sense that the merit function converges to zero along the iterates, which in turn implies that the iterates approach the solution set.

To implement an infeasible interior-point algorithm in the framework of Algorithm 2, we need certain stopping criteria. One natural criterion is

$$\phi^k \le \varepsilon, \tag{8.1}$$

where ε is a prescribed tolerance for optimality. However, when a horizontal linear complementarity problem (2.1) does not have a solution, the merit function will not converge to zero but to a positive value (recall that it is monotonically decreasing). In this case, we use another stopping criterion, which we describe below.

At each iteration, we calculate the quantity

$$\theta^k = \nu^k \frac{(x^0 - u^0)^T y^k + (y^0 - v^0)^T x^k}{x^{k^T} y^k}$$

and terminate Algorithm 2 if

$$\theta^k > 4\beta + 5, \tag{8.2}$$

where β is any positive constant independent of the problem size.

In [8], Kojima, Megiddo, and Mizuno considered the problem of giving certain infeasibility information when an algorithm stops without obtaining an approximate solution. Here we consider the same problem for Algorithm 2 equipped with the stopping criteria (8.1) and (8.2).

THEOREM 8.1. *Let Algorithm* 2 *be equipped with the stopping criteria* (8.1) *and* (8.2), *and let Assumption A*2 *hold* (*Assumption A*1 *may or may not hold*). *Given* $\varepsilon > 0$, *let* $(x^0, y^0) = \rho(e, e)$ *satisfy* (7.6) *and* (7.14). *In at most* $O(n^2 \ln(1/\varepsilon))$ *iterations Algorithm* 2 *terminates at either criterion* (8.1) *or* (8.2). *Moreover, if it terminates at* (8.2), *then there exists no solution* (x^*, y^*) *to Problem* (2.1) *such that* $\|(x^*, y^*)\| \leq \rho\beta\sqrt{n}$.

Proof. It is easy to see that the quantity θ^k is the left-hand side of the inequality in Lemma 6.1.

At each iteration as long as the algorithm does not stop, θ^k is bounded above by $4\beta + 5$, which implies $\omega^k = O(n)$ (see (6.3)–(6.5)). From Theorem 7.3, the algorithm has to stop in at most $O(n^2 \ln(1/\varepsilon))$ iterations.

If Problem (2.1) has a solution (x^*, y^*) such that $\|(x^*, y^*)\| \leq \rho\beta\sqrt{n}$, then (7.7) holds. From the proof of Lemma 7.2, we have $\theta^k \leq 4\beta + 5$. Therefore, the algorithm cannot stop at (8.2). This completes the proof. □

In normal cases, it should be reasonable to choose ε sufficiently small, and $\rho\beta$ sufficiently large.

9. Further comments. In this section, we briefly summarize the results of the current work, and then offer some relevant comments on the important issues of complexity, finite termination, and local convergence.

9.1. Summary. Infeasible interior-point algorithms in the framework of the centered and damped Newton method empirically have proved to be among the most efficient interior-point algorithms for linear programming. But until now, their theoretical convergence properties were not well understood. In this paper, we resolved the convergence issue for an infeasible interior-point algorithm from this framework with a particular steplength control procedure, and also established an $O(n^2 \ln(1/\varepsilon))$-iteration complexity bound for the algorithm to attain feasibility and complementarity within the ε accuracy. No significant restriction was imposed upon the algorithmic framework, except for a quite liberal control on the steplength selection.

9.2. Complexity. The study of worst-case complexity has been one of the focal points of research on interior-point methods. In the meantime, however, the inadequacy of worst-case complexity as a measure of computational efficiency has been

long and increasingly acknowledged. Currently, the best measure of efficiency is still numerical performance. It is unfortunate that worst-case complexity bounds obtained for noncombinatorial algorithms, including both feasible interior-point and infeasible interior-point ones, are often too conservative or too far from average cases. As a result, it seems only appropriate to use worst-case complexity as a qualitative, rather than quantitative, measure of reliability. An algorithm with a polynomial complexity bound is likely more reliable than one without; but for two polynomial algorithms, a small difference in their presently available worst-case complexity bounds provides, in our opinion, little or no information as to which one of the two is likely to be more reliable or more efficient. Based on this belief, we do not consider it of practical value to pursue a slightly better complexity bound at the price of imposing more stringent restrictions on an otherwise practically efficient algorithm. This was precisely the reason behind our effort to keep the algorithm analyzed as close to the algorithm implemented as we can.

9.3. Finite termination. It is well known that in theory interior-point methods for linear programming can be terminated in a finite number of iterations thanks to the combinatorial nature of linear programming. Recently, Ye [21], and Mehrotra and Ye [16] proposed and studied effective finite termination procedures for primal-dual interior-point linear programming algorithms. Following the similar line of analysis, one can show that for a given linear program, an infeasible interior-point approximate solution (x^k, y^k) satisfying ε-feasibility and ε-complementarity, as well as condition (5.6), will eventually provide the correct zero-nonzero partition when ε is small enough. Therefore, effective finite termination procedures can also be applied to infeasible interior-point algorithms.

9.4. Local convergence. An important aspect of interior-point methods is their local behavior. The analysis of local convergence rate for primal-dual interior-point methods has recently attracted considerable research interest. At this point, it is natural to ask whether or not infeasible interior-point methods can also achieve superlinear and quadratic convergence as their interior-point counterparts. So far, ample empirical evidence suggests that the answer should be in the affirmative. In [25], Zhang, Tapia, and Potra already gave sufficient conditions for infeasible interior-point methods in the framework of Algorithm 1 to attain superlinear convergence. These conditions are extensions of those for feasible interior-point methods (see [24] and [23]); basically, ensuring $\sigma^k \to 0$ and $\alpha^k \to 1$. Although it does not seem particularly difficult to establish a fast convergence rate for infeasible interior-point algorithms under the assumption of nondegeneracy or convergence of the iteration sequence, it is of more interest without these assumptions. This is currently a subject of further investigation.

Acknowledgment. I am indebted to Richard Tapia for his persistence in encouraging me to carry through this investigation, especially at times when the outcome did not appear particularly attractive. I am also grateful to M. Seetharama Gowda for his observation that led to Proposition 3.1, and to an anonymous referee for constructive comments and suggestions.

REFERENCES

[1] K. M. ANSTREICHER, *A combined phase* I *– phase* II *projective algorithm for linear programming*, Math. Programming, 43 (1989), pp. 209–223.

[2] K. M. ANSTREICHER, *A combined phase* I – *phase* II *scaled potential algorithm for linear programming*, Math. Programming, 52 (1991), pp. 429–440.

[3] R. W. COTTLE, J. S. PANG, AND R. E. STONE, *The linear complementarity problem*, Academic Press, Boston, 1992.

[4] G. DE GHELLINCK AND J.-PH. VIAL, *A polynomial Newton method for linear programming*, Algorithmica, 1 (1986), pp. 425–453.

[5] A. V. FIACCO AND G. P. MCCORMICK, *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*, John Wiley, New York 1968 and SIAM, Philadelphia, 1990.

[6] O. GÜLER, *Generalized Linear Complementarity Problems and Interior point Algorithms for Their Solutions*, Tech. Report, Faculty of Technical Mathematics and Computer Science, Delft University of Technology, the Netherlands, April 1992.

[7] T. ISHIHARA AND M. KOJIMA, *On the Big $\mathcal{M}$ in the Affine Scaling Algorithm*, Res. Report on Information Sciences, B-255, Dept. of Information Sciences, Tokyo Institute of Technology, Tokyo, Japan, 1992.

[8] M. KOJIMA, N. MEGIDDO, AND S. MIZUNO, *A Primal-Dual Exterior Point Algorithm for Linear Programming*, Res. Report RJ 8500, IBM Almaden Research Center, San Jose, CA, 1991.

[9] M. KOJIMA, N. MEGIDDO, AND T. NOMA, *Homotopy continuation methods for complementarity problems*, Math. Oper. Res., 16 (1991), pp. 754–774.

[10] M. KOJIMA, S. MIZUNO, AND A. YOSHISE, *A Little Theorem of the Big $\mathcal{M}$ in Interior Point Algorithms*, Res. Report on Information Sciences, B-239, Dept. of Information Sciences, Tokyo Institute of Technology, Tokyo, Japan, 1991.

[11] I. J. LUSTIG, R. E. MARSTEN, AND D. F. SHANNO, *Computational experience with a primal-dual interior point method for linear programming*, Linear Algebra Appl., 152 (1991), pp. 191–222.

[12] ———, *On implementing Mehrotra's predictor-corrector interior point method for linear programming*, SIAM J. Optimization, 2 (1992), pp. 435–449.

[13] K. A. MCSHANE, C.L. MONMA, AND D.F. SHANNO, *An implementation of a primal-dual interior point method for linear programming*, ORSA J. Computing, 1 (1989), pp. 70–83.

[14] N. MEGIDDO, *Pathways to the optimal set in linear programming*, in Progress in Mathematical Programming, Interior-Point and Related Methods, N. Megiddo, ed., Springer-Verlag, New York, 1989, pp. 131–158,

[15] S. MEHROTRA, *On the implementation of a primal-dual interior point method*, SIAM J. Optimization, 2 (1992), pp. 575–601.

[16] S. MEHROTRA AND Y. YE, *On finding the optimal facet of linear programs*, Tech. Report, Department of IE and MS, Northwestern University, Evanston, IL, 1991.

[17] M. J. TODD, *On Anstreicher's Combined Phase* I–*Phase* II *Projective Algorithm for Linear Programming*, Tech. Report 776, School of Operations Research and Industrial Engineering, Cornell University, Ithaca, NY, 1989; Math. Programming, to appear.

[18] M. J. TODD AND Y. WANG, *On Combined Phase* I–*Phase* II *Projective Methods for Linear Programming*, Tech. Report 877, School of Operations Research and Industrial Engineering, Cornell University, Ithaca, NY, December 1989; Algorithmica, to appear.

[19] Q. XU, *On the New Linear Programming Algorithms—New Sequential Penalty Function Method and Two Point Barrier Function Method*, Ph.D. thesis, Institute of Nuclear Technology, Tsinghua University, Beijing, China, 1991. (In Chinese.)

[20] Y. YE, *A Low Complexity Combined Phase* I–*Phase* II *Potential Reduction Algorithm for Linear Programming*, Working Paper Series No. 91-1, College of Business Administration, University of Iowa, Iowa City, IA, 1991.

[21] ———, *On the Finite Convergence of Interior-Point Algorithms for Linear Programming*, Working Paper 91-5, the College of Business Administration, University of Iowa, Iowa City, IA, 1991.

[22] Y. ZHANG AND R. A. TAPIA, *A superlinearly convergent polynomial primal-dual interior-point algorithm for linear programming*, SIAM J. Optimization, 3 (1993), pp. 118–133.

[23] ———, *Superlinear and quadratic convergence of primal-dual interior point algorithms for linear programming revisited*, J. Optim. Theory Appl., 73 (1992), pp. 229–242.

[24] Y. ZHANG, R. A. TAPIA, AND J. E. DENNIS, JR., *On the superlinear and quadratic convergence of primal-dual interior point linear programming algorithms*, SIAM J. Optimization, 2 (1992), pp. 304–324.

[25] Y. ZHANG, R. A. TAPIA, AND F. POTRA, *On the superlinear convergence of interior point algorithms for a general class of problems*, SIAM J. Optimization, 3 (1993), pp. 413–422.

SIAM J. OPTIMIZATION
Vol. 4, No. 2, pp. 229–246, May 1994
© 1994 Society for Industrial and Applied Mathematics

LINE SEARCH PROCEDURES FOR THE LOGARITHMIC BARRIER FUNCTION*

WALTER MURRAY† AND MARGARET H. WRIGHT‡

Abstract. Barrier methods for constrained optimization, widely applied in the 1960s and 1970s, have recently enjoyed a revival of popularity. A line search with respect to a logarithmic barrier function or related potential function is often part of modern interior methods. Standard line search procedures tend to be inefficient in this context for two reasons: an inappropriate choice of initial trial step if any constraints are nonlinear, and poor approximation of the barrier function by low-order polynomial interpolants. This paper discusses line search strategies specifically designed for the logarithmic barrier function.

Key words. barrier function, line search, interior methods

AMS subject classifications. 65K05, 90C30

1. Barrier methods for constrained optimization.

Starting in 1984 with the work of Karmarkar [KA84], a resurgence of interest has taken place in barrier methods for constrained optimization. Classical barrier methods, which were popular in the 1960s and early 1970s, treat inequality constraints by creating a transformed function containing a positive singularity ("barrier") at the boundary of the feasible region.

Consider the constrained optimization problem

$$\underset{x\in\Re^n}{\text{minimize}}\ f(x) \quad \text{subject to} \quad c_i(x) \geq 0,\ \ i = 1,\ldots,m. \tag{1}$$

We assume that the objective function f and the set of constraint functions $\{c_i\}$ are smooth. A point $\bar{x}$ for which $c_i(\bar{x}) > 0$, $i = 1, \ldots, m$, is said to be *strictly feasible.*

Let x^* denote a local solution of (1). The corresponding logarithmic barrier function is

$$B(x,\mu) = f(x) - \mu \sum_{i=1}^{m} \ln c_i(x), \tag{2}$$

where μ is a positive scalar called the *barrier parameter.* Note that this barrier function is defined only at strictly feasible points. Let $x(\mu)$ denote an unconstrained minimizer of $B(x,\mu)$. Under certain conditions, it can be shown that

$$\lim_{\mu\to 0} x(\mu) = x^*.$$

For details about background and theory, see [FM68], [FI79], and [WR92].

Most interior methods proposed since 1984 are designed for the special cases of linear programming (LP) and convex quadratic programming, and the vast majority of papers on these topics have concentrated on proofs of polynomial complexity (see the extensive bibliographies in [GT89] and [GO92]). Within the last few years, several

* Received by the editors February 7, 1992; accepted for publication (in revised form) January 6, 1993.

† Operations Research Department, Stanford University, Stanford, California 94305 (walter@sol-walter.stanford.edu). The work of this author was supported by U.S. Department of Energy grant DE-FG03-92ER25117, National Science Foundation grant DDM-8715153 and Office of Naval Research grant N00014-90-J-1242.

‡ AT&T Bell Laboratories, Murray Hill, New Jersey 07974 (mhw@research.att.com).

interior methods for convex nonlinear programming have been proposed and analyzed from the viewpoint of complexity; see, for example, [NN89], [DRT], [JA90], and [MS90]. On the practical side, barrier techniques are increasingly being applied to nonlinearly constrained problems of many varieties, including nonconvex and large-scale instances; see, for example, [JS91], [NS93], and [RI91].

Barrier methods fell from favor during the 1970s partly because inherent ill-conditioning in the Hessian matrices created difficulties for standard unconstrained methods. A major issue today is the development of strategies for dealing with this ill-conditioning; for a discussion of some approaches, see [WR92]. A second inconvenient feature of barrier functions is that they tend to cause inefficiencies for general-purpose line search techniques. In the next section, we discuss the role of a line search in optimization algorithms, and explain why special-purpose line search procedures may be helpful in dealing with barrier functions.

The line search procedures to be described apply to unconstrained minimization of a sequence of barrier functions (2) with decreasing values of μ, and also to linearly and nonlinearly constrained subproblems in which a barrier function serves as a merit function.

2. Line searches for barrier functions.

2.1. The role of the line search. In many optimization methods, a typical iteration (say, the kth) defines a new iterate x_{k+1} by taking a positive step α_k from the current point x_k along an n-dimensional search direction p_k, so that

$$x_{k+1} = x_k + \alpha_k p_k. \tag{3}$$

We shall assume henceforth that p_k is a descent direction for some differentiable function Φ, i.e. that $p_k^T \nabla\Phi(x_k) < 0$. The process of choosing α_k is usually called the *line search.*

Iterations of the form (3) arise in several contexts. In line search methods for unconstrained minimization of Φ, α_k is required to satisfy mathematical conditions of *sufficient decrease* with respect to properties of Φ at x_k (see, for example, [OR70], [DS83], [DS89], [FL87], and [GMW]). Sufficient decrease conditions require more than simply a reduction in Φ; they demand that Φ decrease "enough" to satisfy certain requirements for convergence of $\{x_k\}$ (see [OR70] for detailed proofs).

Given x_k and p_k, let $\phi(\alpha)$ denote the function

$$\phi(\alpha) = \Phi(x_k + \alpha p_k), \tag{4}$$

so that $\phi(\alpha)$ represents the variation in Φ as a function of the step α from x_k along p_k. Then

$$\phi(0) = \Phi(x_k), \quad \phi'(0) = p_k^T \nabla\Phi(x_k) \quad \text{and} \quad p_k^T \nabla\Phi(x_k + \alpha p_k) = \phi'(\alpha).$$

Because p_k is a descent direction for Φ, it follows that $\phi'(0) < 0$. Two frequently used sufficient decrease conditions are posed in terms of constants σ_1 and σ_2 satisfying $0 < \sigma_1 < \sigma_2 < 1$: the step α_k is accepted only if

$$\phi(\alpha_k) < \phi(0) + \sigma_1 \alpha_k \phi'(0) \quad \text{and} \quad \phi'(\alpha_k) \geq \sigma_2 \phi'(0). \tag{5}$$

For other sufficient decrease conditions, see, for example, [OR70], [BN89], [DS83], [DS89], [FL87], and [GMW].

Some algorithms for constrained optimization include a line search in the sense that they require a sufficient decrease at each iteration in a merit function intended to measure progress toward the solution. Although the definition of the merit function may vary between iterations, sufficient decrease conditions such as (5) are applied during any given iteration. For simplicity, $\Phi(x)$ and the associated one-variable function $\phi(\alpha)$ will always denote the generic functions in which a sufficient decrease is sought.

2.2. The line search iteration. A suitable value of α_k is usually found by an iterative procedure that generates a sequence of trial steps $\{\alpha^{(j)}\}$ for $j = 0, \ldots$, and terminates by accepting the first $\alpha^{(j)}$ for which the sufficient decrease conditions hold. A certain amount of work is associated with calculating and testing each trial point $\alpha^{(j)}$. For example, simply checking the sufficient decrease conditions (5) requires evaluation of Φ and $\nabla\Phi$ at $x_k + \alpha^{(j)}p_k$.

The choice of the initial step $\alpha^{(0)}$ in the line search almost always depends on the outer strategy that defines p_k. For Newton and quasi-Newton methods (the only methods considered here) the obvious "natural" initial step of unity ($\alpha^{(0)} = 1$) eventually produces the correct convergence rate as well as a sufficient decrease for iterates near the solution (see [DM77]).

In the early years of line search methods for unconstrained optimization, most implementations included an "accurate" line search, i.e., α_k was a close approximation to a minimizer of ϕ. Today, in contrast, it is customary to perform only an "inexact" or "relaxed" line search, typified by a small value of σ_1 and a large value of σ_2 in (5) (say, $\sigma_1 = 10^{-4}$ and $\sigma_2 = 0.9$). An accurate line search nonetheless remains desirable in some situations, such as in large-scale problems where the linear algebra needed to compute p_k may be substantially more time-consuming than evaluation of Φ at trial steps. See [DS83], [FL87], and [GMW] for additional discussion.

Typical relaxed sufficient decrease criteria are satisfied by a *range* of acceptable values for α_k. An effective line search should therefore produce a "good" step (broadly speaking, a large reduction in Φ) without requiring too many trial steps. The tradeoff is somewhat delicate, and the optimal balance is problem-dependent.

In the popular *Armijo* or *pure backtracking* line search ([OR70], [DS83]), each trial step $\alpha^{(j+1)}$, $j = 0, \ldots$, is taken as a fixed fraction τ (typically $\tau = \frac{1}{2}$ or $\tau = \frac{1}{10}$) of the previous step $\alpha^{(j)}$. Under reasonably mild conditions, this procedure is guaranteed to produce a sufficient decrease (see, for example, [BN89]). The great advantage of an Armijo line search is its simplicity. In instances where the initial step is almost always accepted—for instance, in minimizing well-behaved functions with Newton's method—an Armijo line search is also efficient. However, a pure backtracking procedure is inappropriate when an accurate line search is needed, since the fixed, strictly decreasing sequence $\{\tau^j\alpha^{(0)}\}$, $j = 1, \ldots$, of trial steps is unaffected by the behavior of Φ along p_k.

A second standard line search technique is to define the trial steps as iterates from a procedure for (approximately) minimizing $\phi(\alpha)$ within a specified interval of uncertainty. For unconstrained problems, the initial interval of uncertainty is bounded on the left by $\alpha = 0$ (where $\phi' < 0$) and on the right by (usually) a large step that represents the maximum permitted change in x during a particular iteration. The interval of uncertainty is modified to reflect information about ϕ accumulated as the line search proceeds. For example, if $\phi'(\alpha^{(0)})$ is positive, at least one minimizer must occur in the interval $(0, \alpha^{(0)})$ (in which case we say that the minimizer is *bracketed*). Furthermore, any positive value of ϕ' satisfies the second sufficient decrease condition

in (5). It should be stressed that univariate minimization is used only as a model subproblem for generating trial steps in the line search; the sequence $\{\alpha^{(j)}\}$ terminates as soon as the desired sufficient decrease conditions are satisfied.

For smooth functions, low-order polynomial interpolation is widely used to construct $\alpha^{(j+1)}$ based on previous trial steps and the associated values of ϕ (see [BR73], [DS83], [DS89], [FL87], [GM74], and [GMW]). This approach is sometimes combined with backtracking (see, for example, [BN89] and [DS89]), and can be used when an accurate line search is desired. To ensure robustness, the interpolant step α_{int} can be rejected under certain circumstances; for example, if α_{int} fails to lie within the current interval of uncertainty, a safeguarded step that produces a guaranteed reduction in the interval of uncertainty is used instead. This issue and related topics are discussed in [BR73], [DS89], [FL87], and [GM74].

In many implementations of general-purpose line search methods, the choice of interpolating polynomial depends in part on the information available to the outer optimization algorithm. In particular, when p_k is obtained from a Newton-based method in which a second-derivative matrix is calculated or approximated by finite differences at every iterate, it is implicitly assumed that computation of the gradient is not too expensive relative to that of the function. Hence the interpolating function in the associated line search is typically a *cubic* function chosen to match ϕ and ϕ' at two trial steps. Since ϕ' is the directional derivative of Φ along p_k, strictly speaking the full gradient vector $\nabla\Phi$ is not needed to obtain ϕ', which could be approximated by, for example, a single finite-difference of Φ along p_k. In practice, however, most optimization software calculates ϕ' by forming the inner product of the vectors $\nabla\Phi$ and p_k.

When calculation of the gradient $\nabla\Phi$ is very expensive relative to that of Φ, it is standard to compute $\nabla\Phi$ (or its finite-difference approximation) only at each outer iterate x_k, and not at trial steps $x_k + \alpha^{(j)}p_k$ during the line search. In this situation, common practice is to perform interpolation with a *quadratic* function fitted to data in two forms. For the first interpolation during a given line search, a quadratic is matched to $\phi(0)$, $\phi'(0)$, and $\phi(\alpha^{(0)})$; the value of $\phi'(0)$ is assumed to be available, since the vector $\nabla\Phi(x_k)$ is needed to define p_k. After the first interpolation, a quadratic polynomial is fitted to three function values. Although cubic-based interpolation schemes can be developed for this case, quadratic interpolation has been the most widely implemented procedure; see, for example, the line search routines in the Numerical Algorithms Group library [NAG]. Some of the issues in choosing a parabolic or cubic fit are described in [GMW, §8.1.3.2]; several related topics are discussed in [FL87, §2.6].

2.3. Specialized line searches for barrier functions. In the 1960s and 1970s, the first heyday of barrier methods, several special-purpose line search strategies were proposed. In [FLMC], separate approximations are made to the objective function and the barrier term. In the approach of [LFR], a special function is developed by substituting linearized versions of the objective function and constraints into the form of the barrier function. The techniques of both [FLMC] and [LFR] apply to a broad class of barrier functions. The approximating functions of §4 (first suggested in [WR76] and [MW76]) are designed specifically for the logarithmic barrier function.

Despite the great interest in barrier methods since 1984, specialized line search strategies have received very little attention in recent papers, primarily because an extremely simple line search appears to be entirely adequate for the predominant case of linear programming. The typical line search in barrier-related linear programming

methods chooses α_k in almost all circumstances as a fixed fraction close to unity (such as 0.95 or 0.99) of the step from x_k along p_k to the boundary of the feasible region. (See the discussion and references in [Go92].) Broadly speaking, this strategy succeeds for the LP case because in many instances singularity at the boundary causes a minimizer to occur near the boundary in a function that would otherwise behave like a monotonically decreasing linear function.

For nonlinear problems, however, such a simple approach tends to be inefficient, as we shall discuss in §3.1. Furthermore, specialized line search techniques may be helpful in locating points that satisfy certain conditions imposed on the line search to obtain desirable complexity properties for convex problems; see, for example, [ADRT] and [AN90]. A recent line search strategy designed to exploit the analyticity of the barrier function is described in [JA92].

For minimizing the log barrier function, we suggest modifications to two aspects of a general-purpose line search: (i) the choice of initial step $\alpha^{(0)}$; and (ii) the procedure for generating subsequent trial steps. With respect to the first, choosing $\alpha^{(0)}$ to retain strict feasibility poses no difficulty when *all* the constraints are linear (as in linear and quadratic programming), since the nonnegative step along p_k to the boundary of the feasible region ("the step to the nearest constraint") can be computed directly, and then imposed as the right endpoint of the interval of uncertainty. When even a single constraint is nonlinear, the situation is more complicated. Selection of the initial step will be discussed in §3.

The second issue arises because low-order polynomial interpolants may fail to reflect the characteristic dramatic behavior of a barrier function in a neighborhood of the singularity. The disparity in an example with $\mu = \frac{1}{2}$ is illustrated in Fig. 1. Special-purpose interpolating functions containing a logarithmic singularity are described in §4.

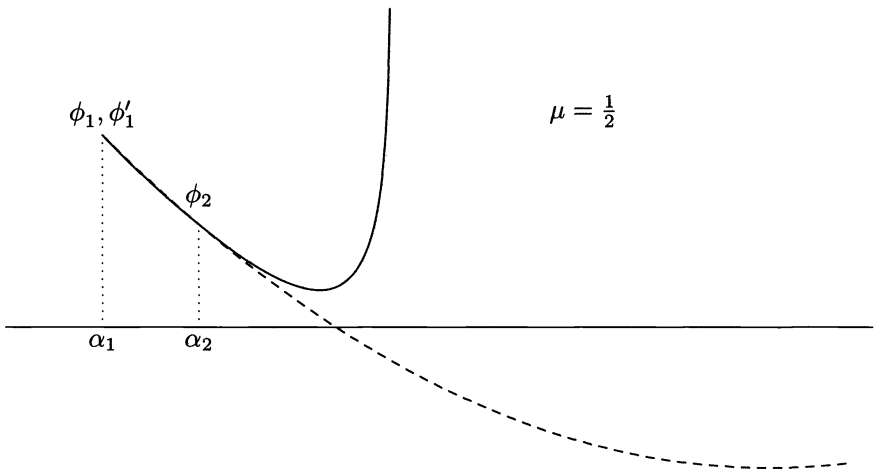


FIG. 1. *A barrier function $\phi(\alpha)$ with barrier parameter $\mu = \frac{1}{2}$ is shown as a solid line. The dashed line represents the quadratic function that interpolates $\phi(\alpha_1)$, $\phi'(\alpha_1)$, and $\phi(\alpha_2)$. The minimizer of the quadratic (at the extreme right of the figure) is a substantial and infeasible overestimate of the minimizer of the barrier function, even for this relatively large value of μ.*

3. Choosing the initial trial step.

3.1. Difficulties arising from constraint nonlinearity. Before discussing our approach to choosing the initial trial step, we stress that the issue is *not* simply retention of strict feasibility; if it were, a fixed tiny step along p_k would almost always suffice. As noted in §2.2, a practical line search strategy should retain some level of efficiency throughout the overall minimization. In particular, it is desirable to take the natural initial step of unity near the solution. If this step is infeasible, an Armijo-like strategy of trying fixed multiples of the initial step will waste constraint evaluations in the frequent situation when the step to the boundary of the feasible region is much smaller than unity.

The difficulties arising from constraint nonlinearities are revealed by considering the following hypothetical generalization of the linear-constraint strategy: calculate γ_k, the smallest positive step such that some constraint vanishes at $x_k + \gamma_k p_k$; then choose the right endpoint of the initial interval of uncertainty as a value slightly less than γ_k. Although this idealized strategy would retain feasibility, its implementation is problematic for two reasons.

First, the question arises of how to calculate γ_k. For nonlinear constraints, γ_k must be determined iteratively—for example, by applying a numerical zero-finding procedure to compute all positive steps along p_k at which any constraint vanishes, and then taking γ_k as the smallest of these values. Since an arbitrary nonlinear function may "wiggle," however, a generic zero-finding procedure will not necessarily locate the *smallest* positive step along p_k to a constraint zero. Thus there is no guarantee that the correct value of γ_k has been obtained.

Second, the expense of trying to obtain an accurate value of γ_k is not necessarily worthwhile. Except for the special case of linear programming, there is no reason to expect a barrier function to achieve its minimum close to the boundary. In fact, a longstanding rule of thumb for nonlinear problems is that a premature move too close to the boundary can lead to a long sequence of iterates "trapped" near the singularity by the need to reduce the barrier function at each iteration (see, for example, [FLMC], [WR76]). Since accurate calculation of γ_k is therefore unlikely to be cost-effective as a general strategy, we consider how the available information at $\alpha = 0$ can be applied to make a sensible choice for the initial step.

3.2. A computational procedure. The discussion in this section involves a single line search and we accordingly drop the subscript k, so that x denotes the current iterate, and so on. Let α_u denote the standard choice of initial step for the algorithm used to define p. (For Newton and quasi-Newton methods, $\alpha_u = 1$.) To preserve the asymptotic rate of convergence, the step α_u should be taken whenever it produces a sufficient decrease. However, the step α^* to the minimizer of the barrier function may be much less than α_u because of a nearby singularity whose effect is masked at x by a small value of μ. In this section, a strategy is presented whose intent is to recognize this situation and then, if necessary, to compute an estimate α_b of α^*.

The first ingredient in the process is an easily computed estimate $\bar{\alpha}$ of the step to the nearest constraint. For each constraint j that is decreasing along p, we compute σ_j, the (necessarily positive) step to the zero of a linearized version of the constraint:

$$\sigma_j = -\frac{c_j}{a_j^T p} \quad \text{for} \quad a_j^T p < 0. \tag{6}$$

If $a_j^T p \geq 0$ for all j, we define $\bar{\alpha} = +\infty$. Otherwise, $\bar{\jmath}$ and $\bar{\alpha}$ are defined as

$$\bar{\alpha} = \sigma_{\bar{\jmath}} = \min \sigma_j, \tag{7}$$

so that $\bar{\jmath}$ is a prediction of the nearest constraint. (The value of $\bar{\alpha}$ is unique, whereas $\bar{\jmath}$ need not be; however, only $\bar{\alpha}$ is required in the suggested procedure.) Calculation of $\bar{\alpha}$ requires the (known) values and gradients of constraints decreasing at x. If all the constraints are linear, $\bar{\alpha}$ is the exact value γ.

When $\bar{\alpha}$ is finite, a strategy analogous to standard practice in linear programming would simply define α_b as a fixed fraction of $\bar{\alpha}$, say as $0.99\bar{\alpha}$. If all the constraints are strictly concave, however, the value of $\bar{\alpha}$ is an *overestimate* of the step to constraint $\bar{\jmath}$, and $x + \bar{\alpha}p$ is infeasible; this suggests caution in choosing the initial step too close to $\bar{\alpha}$ (see the discussion following (11)). Accordingly, we have chosen a marginally more complicated policy, based on the observation that a stationary point α of the barrier function along p satisfies the following nonlinear equation:

$$p^T \nabla B(x + \alpha p, \mu) = p^T g(x + \alpha p) - \sum_{j=1}^{m} \frac{\mu}{c_j(x + \alpha p)} p^T a_j(x + \alpha p) = 0. \tag{8}$$

A good estimate of the solution of (8) can be calculated using already-available information by assuming that (i) only constraint $\bar{\jmath}$ is relevant and (ii) fixed local linear approximations to f and constraint $\bar{\jmath}$ are adequate. (The latter approximations are exact if f and $c_{\bar{\jmath}}$ are linear.) These simplifications lead to the relation

$$g^T p - \frac{\mu}{c_{\bar{\jmath}} + \alpha a_{\bar{\jmath}}^T p} a_{\bar{\jmath}}^T p = 0, \quad \text{which holds at} \quad \alpha = \bar{\alpha} + \frac{\mu}{g^T p}. \tag{9}$$

The step defined by (9) is accepted only if it is both less than $\bar{\alpha}$ and not "too small" relative to $\bar{\alpha}$. To ensure a step less than $\bar{\alpha}$, relation (9) is used only when $g^T p < 0$, i.e., when the objective function itself is decreasing along p. If $g^T p > 0$, the step is taken as a fraction of $\bar{\alpha}$ (say, $0.95\bar{\alpha}$). The decision that the step is "too small" is heuristic; we have chosen $0.5\bar{\alpha}$ as a lower bound.

A step α_b based on the barrier function is thus defined as

$$\begin{aligned} \alpha_b &= \max(\bar{\alpha} + \mu/g^T p,\ 0.5\bar{\alpha}) && \text{if} \quad \bar{\alpha} < +\infty \text{ and } g^T p < 0; \\ &= 0.95\bar{\alpha} && \text{if} \quad \bar{\alpha} < +\infty \text{ and } g^T p \geq 0; \\ &= +\infty && \text{if} \quad \bar{\alpha} = +\infty. \end{aligned} \tag{10}$$

The initial step $\alpha^{(0)}$ in a given line search is then taken as the smaller of α_b and α_u:

$$\begin{aligned} \alpha^{(0)} &= \alpha_b \quad \text{if } \alpha_b < \alpha_u; \\ &= \alpha_u \quad \text{otherwise.} \end{aligned}$$

Although it is theoretically possible that α_b could be less than α_u at every iteration, this situation is extremely unlikely when a superlinearly convergent method is used to minimize the barrier function, since the iterates should eventually tend to lie near the barrier trajectory. Close to x^*, the value of σ_j given by (6) for an *inactive* constraint will typically be much greater than one, since c_j is bounded away from zero and $a_j^T p$ is small. For an *active* constraint j and sufficiently small μ, analysis of the barrier trajectory shows that a step p from $x(\bar{\mu})$ to $x(\mu)$, with $\mu < \bar{\mu}$, approximately satisfies $a_j^T p = -(1 - \mu/\bar{\mu})c_j$; see, e.g., [WR76] and [WR92]. The value of σ_j will thus

tend to be greater than one for all j, which means that $\bar{\alpha}$ will exceed one. For small enough μ, α_b will also be greater than one, and $\alpha^{(0)}$ will be taken as α_u.

To illustrate calculation of the initial trial step, we consider a simple example:

$$\text{(11)} \quad \text{minimize} \quad x_1 x_2^2 \quad \text{subject to} \quad -x_1^2 - x_2^2 + 2 \geq 0 \quad \text{and} \quad -x_1^2 x_2^2 + 10 \geq 0.$$

Note that these particular constraints would more naturally be expressed in "less than or equal" form. The second constraint is redundant, but is included to show that the barrier function reflects all constraints at every iteration.

The numerical results in this paper were calculated in double precision (around 16 decimal digits) on a Silicon Graphics 4D/240S, and have been rounded to four decimal digits. The numbers given correspond to the point $x = (-.81, -1.1)^T$, for which the constraint values are $c_1 = .1339$ and $c_2 = 9.206$.

When the barrier parameter μ is taken as 0.01, the value of the barrier function is -0.9822, and the Newton direction is $p = (-.8602, -1.459)^T$. The exact step γ to the nearest constraint and the exact minimizer α^* of the barrier function along p are

$$\text{(12)} \qquad \gamma = .02857 \qquad \text{and} \qquad \alpha^* = .02601.$$

The quantities σ_j defined by (6) for the two constraints are $\sigma_1 = .02909$ and $\sigma_2 = 2.428$; the value of $g^T p$ is -3.640. Applying (7) and (9), we have

$$\text{(13)} \qquad \bar{\alpha} = \sigma_1 = .02909 \qquad \text{and} \qquad \alpha_b = .02634.$$

Observe that $\bar{\alpha}$ exceeds γ, since the first constraint is concave. Because $\alpha_u = 1$ and $\alpha_b < \alpha_u$, the initial step $\alpha^{(0)}$ is taken as α_b. The values of the constraints for the step α_b are $c_1 = .01065$ and $c_2 = 9.101$, and the value of the barrier function is -1.056.

To ensure feasibility, all nonlinear constraints must be evaluated at every trial step $\alpha^{(j)}$. (It is not necessary to check feasibility for *linear* constraints for any step less than $\bar{\alpha}$.) If any infeasibilities occur at a trial step, the right endpoint of the interval of uncertainty is updated. Secant estimates of zeros of violated constraints can be calculated from the values of the constraint functions at the infeasible point and the most recent feasible step to its left (which may be $\alpha = 0$). The smallest (positive) estimated step to a constraint zero and the corresponding constraint index then play the roles of $\bar{\alpha}$ and $\bar{\jmath}$ in computing a new estimate of the step from (9).

4. Special interpolating functions. The functions to be described in this section are specifically designed for interpolating the logarithmic barrier function. (Similar functions could be developed for other barrier or potential functions.) Minimizers of these special functions need to be calculated by iteration, but the associated computational effort is small. The interpolation procedures can be incorporated into standard line searches based on safeguarded parabolic or cubic interpolation (see, for example, [DS83] or [GM74]) with slight modifications to the structure of a typical iteration; see §5 for further comments.

In this discussion, let $\phi(\alpha)$ denote the barrier function $B(x_k + \alpha p_k, \mu)$ treated as a function of the single variable α, where x_k is the current iterate, p_k is the search direction, and μ is the barrier parameter. We assume that p_k is a descent direction with respect to the barrier function at x_k, so that $\phi'(0) < 0$. Because $\alpha_k \geq 0$, singularity in the barrier function is relevant only if it occurs at some positive step along p_k.

Safeguarded line searches typically maintain an "interval of uncertainty" $[\alpha_{\text{lo}}, \alpha_{\text{hi}}]$ in which a point of sufficient decrease is known to lie; the step α_{best} is the "best"

point found thus far. (For details, see, for example, [BR73] or [GM74].) During an iteration of the line search, it is numerically advantageous to compute an *increment* to the current step rather than a new step itself—for example, by shifting the points so that α_{best} is treated as "zero" (see [GM74]). For simplicity, our discussion in this section involves a variable ζ, an increment to the step, taken as zero at α_{lo}, the *left* end of the current interval of uncertainty; the next trial step is obtained by adding the final value of ζ to α_{lo}. The procedures can be applied in an obvious manner if α_{hi} is the origin for ζ.

In the next two sections, we present two special interpolating functions that can be fitted to the same data used in parabolic or cubic interpolants.

4.1. Linear plus logarithmic singularity. The first interpolating function has the form

$$\ell(\zeta) = a + b\zeta - \mu \ln(d - \zeta),$$

where d is the positive step to the singularity. Because $\ell(\zeta)$ depends on three parameters, it corresponds to a parabolic interpolant. The first and second derivatives of $\ell(\zeta)$ are given by

$$\ell' = b + \frac{\mu}{d - \zeta} \quad \text{and} \quad \ell'' = \frac{\mu}{(d - \zeta)^2}.$$

Because its second derivative is always positive, ℓ is convex. A minimizer of ℓ occurs at ζ_ℓ satisfying

$$\ell'(\zeta_\ell) = b + \frac{\mu}{d - \zeta_\ell} = 0, \quad \text{so that} \quad \zeta_\ell = d + \frac{\mu}{b}. \tag{14}$$

Determination of b and d depends on the data to which $\ell(\zeta)$ is fitted during the line search. If the initial trial step $\alpha^{(0)}$ is not accepted, the first line search iteration is based on fitting $\ell(\zeta)$ to the values ϕ and ϕ' at $\alpha = 0$, and to $\phi(\alpha^{(0)})$; see §4.1.1. Thereafter, unless infeasibility is encountered, $\ell(\zeta)$ is fitted to the three best (usually, the smallest) values of ϕ; see §4.1.2.

4.1.1. Two function values, one first derivative. This situation occurs at the very first iteration of a given line search, when the special function is fitted at $\zeta_1 = \alpha = 0$ and $\zeta_2 = \alpha^{(0)}$. To match ϕ_1 and ϕ_1' at ζ_1 and ϕ_2 at ζ_2, the parameters a, b, and d in $\ell(\zeta)$ must satisfy

$$\phi_1 = a - \mu \ln d, \tag{15a}$$

$$\phi_1' = b + \frac{\mu}{d}, \tag{15b}$$

$$\phi_2 = a + b\zeta_2 - \mu \ln(d - \zeta_2). \tag{15c}$$

Because the barrier function is decreasing at $\alpha = 0$, we know that $\phi_1' < 0$.

Relations (15a) and (15b) give

$$a = \phi_1 + \mu \ln d \quad \text{and} \quad b = \phi_1' - \frac{\mu}{d}.$$

Substituting these values into (15c), we obtain

$$\phi_2 = \phi_1 + \mu \ln d + \zeta_2 \phi_1' - \frac{\zeta_2 \mu}{d} - \mu \ln(d - \zeta_2).$$

Rearrangement produces a nonlinear equation satisfied by d:

$$\ln\left(\frac{d-\zeta_2}{d}\right)+\frac{\zeta_2}{d}+k_1=0, \tag{16}$$

where

$$k_1=\frac{1}{\mu}(\phi_2-\phi_1-\zeta_2\phi_1'). \tag{17}$$

Since $\ell'(0)<0$ and $\ell''(\zeta)>0$ for all ζ, the minimizer of ℓ must lie in the interval $(0,d)$. Making the change of variable $y=\zeta_2/d$, so that $0<y<1$ for $d>\zeta_2$, the equation (16) becomes

$$\xi_1(y)=y+\ln(1-y)+k_1=0. \tag{18}$$

It is straightforward to see that (i) $\xi_1(0)=k_1$ and $\lim_{y\to 1-}\xi_1(y)=-\infty$, and (ii) $\xi_1'<0$ and $\xi_1''<0$ for y in $(0,1)$. The equation $\xi_1(y)=0$ therefore has a unique zero in $(0,1)$ if $k_1>0$. The condition $k_1>0$ holds if ϕ_2 lies above the line through ϕ_1 with slope ϕ_1', i.e., the barrier function is higher at ζ_2 than its linear interpolant at ζ_1. If this is not the case, the effect of the singularity is not locally revealed, ℓ is a poor approximation to the barrier function, and safeguarded quadratic interpolation should be used.

Exponentiation and rearrangement of (18) give the simpler nonlinear equation

$$\psi_1(y)=y+e^{-y}e^{-k_1}-1=0. \tag{19}$$

The function ψ_1 is nearly linear, and Newton's method applied to find its zero accordingly converges very rapidly. Because the effects of y and k_1 are separated, the expression $y+\ln(1-y)$ can be tabulated in advance for a small selection of values of y in $(0,1)$. Once k_1 is calculated, a good starting point for the Newton iteration can be found by table lookup.

Consider example (11), with $\zeta_1=0$ and $\zeta_2=\alpha^{(0)}=\alpha_b=.02634$. (The choice of α_b is explained in §3.) The data to be matched by $\ell(\zeta)$ are

$$\phi_1=-.9822,\quad \phi_1'=-3.292,\quad \text{and}\quad \phi_2=-1.056.$$

The value of k_1 defined by (17) is 1.312. Starting with $y=0.9$ (chosen by table lookup), a single Newton iteration produces $y=.8893$, with $\psi_1=6.279\times 10^{-6}$. We obtain $d=.02962$, $b=-3.630$, and (from (14)) $\zeta_\ell=.02687$. The value of ζ_ℓ is an overestimate of the exact minimizer $\alpha^*=.02601$, but is less than $\gamma=.02857$ and hence feasible (see (12)). The minimizer of the interpolating quadratic fitted to the same data is the *infeasible* step $\alpha=.08710$, so that $\ell(\zeta)$ gives a much better estimate of α^* for this example.

4.1.2. Three function values. Given three function values ϕ_i at steps α_i, $i=1,2,3$, the equations satisfied by a, b, and d are

$$\phi_i=\ell(\zeta_i)=a+b\zeta_i-\mu\ln(d-\zeta_i),\quad i=1,2,3, \tag{20}$$

where $\alpha_3>\alpha_2>\alpha_1$ and $\zeta_i=\alpha_i-\alpha_1$ (so that $\zeta_1=0$).

Elimination of a and b from the three equations in (20) leads to a nonlinear equation satisfied by d:

$$\frac{1}{\zeta_3}\ln\left(\frac{d-\zeta_3}{d}\right)-\frac{1}{\zeta_2}\ln\left(\frac{d-\zeta_2}{d}\right)+k_2=0, \tag{21}$$

where

$$k_2 = \frac{1}{\mu}\left(\frac{\phi_3 - \phi_1}{\zeta_3} - \frac{\phi_2 - \phi_1}{\zeta_2}\right). \tag{22}$$

Let $\nu = \zeta_3/\zeta_2$ and $y = \zeta_3/d$, so that $\nu > 1$ and $0 < y < 1$ for $d > \zeta_3$. Multiplying (21) by ζ_3 and rewriting in terms of y and ν, we obtain

$$\xi_2(y) = \ln(1-y) - \nu \ln\left(1 - \frac{y}{\nu}\right) + k_2\zeta_3 = 0.$$

It is straightforward to show that (i) $\xi_2(0) = k_2\zeta_3$ and $\lim_{y\to 1_-} \xi_2(y) = -\infty$; and (ii) $\xi_2'(y) < 0$ and $\xi_2(y)'' < 0$ for all y in $(0,1)$. These properties imply that ξ_2 has a unique zero in $(0,1)$ if $k_2 > 0$. The condition $k_2 > 0$ means that ϕ_3 lies above the line connecting the points (ζ_1, ϕ_1) and (ζ_2, ϕ_2), so that the barrier function at ζ_3 has increased more rapidly than the linear interpolant of ϕ between ζ_1 and ζ_2. If $k_2 \le 0$, $\ell(\zeta)$ should not be used as an interpolant in the line search.

The following equivalent and somewhat simpler equation is obtained by rearrangement and exponentiation:

$$\psi_2(y) = 1 - y - \beta\left(1 - \frac{y}{\nu}\right)^{\nu} = 0,$$

where $\beta = e^{-k_2\zeta_3}$. The unique zero of ψ_2 in $(0,1)$ can be determined using Newton's method. It is not convenient to find an initial Newton iterate from a table because of the dependence on both β and ν. However, the Newton iterations for ψ_2 can safely be initiated with the starting value $y = 1 - \beta$. The parameters in $\ell(\zeta)$ can then be computed from the relations

$$d = \zeta_3/y, \quad a = \phi_1 + \mu \ln d, \quad \text{and} \quad b = \frac{1}{\zeta_2}\left(\phi_2 + \mu \ln(d - \zeta_2) - a\right),$$

and the minimizer ζ_ℓ is given by (14).

Consider example (11), with the three values of ζ taken as $\alpha_1 = \zeta_1 = 0$, $\zeta_3 = \alpha_b$ (see (13)), and $\zeta_2 = \frac{1}{2}\zeta_3$, for which $\phi_2 = -1.025$. The value of k_2 defined by (22) is 43.80, and $\beta = .3154$. Starting with $y = .6846$, two iterations of Newton's method applied to $\psi_2(y)$ give $y = .9056$ and $\psi_2 = -2.435 \times 10^{-6}$. The parameters of $\ell(\zeta)$ are

$$d = .02909, \quad a = -1.018, \quad \text{and} \quad b = -3.233.$$

Applying relation (14), we obtain $\zeta_\ell = .02638$, which is a (feasible) overestimate of the exact value $\alpha^* = .02601$ (see (12)). The minimizer of the quadratic function fitted to the same three points is the infeasible step .05519.

4.2. Quadratic plus logarithmic singularity. The second interpolating function has the form

$$q(\zeta) = a + b\zeta + c\zeta^2 - \mu \ln(d - \zeta).$$

Since $q(\zeta)$ contains four parameters, it is the analogue of a cubic polynomial in standard line searches. In general, the change from a parabolic to cubic interpolant adds complications because of the need to choose the correct stationary point of the cubic (see, for example, [GM74]). A similar phenomenon occurs when moving from $\ell(\zeta)$ to the more complex function $q(\zeta)$.

The interpolant $q(\zeta)$ is used only when the barrier function is decreasing at the right endpoint of the interval of uncertainty. If the barrier function is increasing at the right endpoint, its minimizer is necessarily bracketed, and a cubic interpolant is likely to provide a reasonable fit; in addition, a minimizer of the special function $q(\zeta)$ is not guaranteed to fall within the interval of uncertainty in this case.

4.2.1. Finding the correct root. Differentiating q, we obtain

$$q' = b + 2c\zeta + \frac{\mu}{d-\zeta}, \quad q'' = 2c + \frac{\mu}{(d-\zeta)^2}, \quad q''' = \frac{2\mu}{(d-\zeta)^3}.$$

Because this interpolant is used only when $q'(0)$ is negative, it follows that $bd+\mu < 0$, and hence that $b < 0$. Since q is unbounded above as $\zeta \to d$, a minimizer of q must exist in the interval $(0, d)$.

A stationary point ζ_q of q occurs when $q' = 0$, which means that ζ_q satisfies the quadratic equation

$$2c\zeta^2 + (b - 2cd)\zeta - \mu - bd = 0. \tag{23}$$

If $c = 0$, so that there is no quadratic term, $q(\zeta)$ reduces to the linear-log function $\ell(\zeta)$ (§4.1).

If $c \neq 0$, there are two roots r_1 and r_2 whose product satisfies

$$r_1 r_2 = \frac{-(bd+\mu)}{2c}. \tag{24}$$

The values of r_1 and r_2 are given by the usual quadratic formula, expressed in two forms:

$$r_1, r_2 = \frac{2cd - b \pm \sqrt{(b-2cd)^2 + 8c(\mu+bd)}}{4c} \tag{25a}$$

$$= \frac{2cd - b \pm \sqrt{(b+2cd)^2 + 8\mu c}}{4c}. \tag{25b}$$

If $c > 0$, both roots are real (from (25b)) and have the same sign (from (24)). Further,

$$+\sqrt{(b+2cd)^2 + 8\mu c} > |b+2cd| = \max(b+2cd, -b-2cd). \tag{26}$$

If $b+2cd > 0$, the combination of (25b) and (26) shows that r_1 (corresponding to the positive square root) exceeds d, whereas $0 < r_2 < d$. If $b+2cd < 0$, then $d < -b/(2c)$; manipulation of (25b) again shows that $r_1 > d$ and $0 < r_2 < d$.

If $c < 0$, relation (25a) reveals that there are two real roots (since $\mu + bd < 0$), and relation (24) shows that the roots are of opposite sign. Because of division by the negative value c, relation (25a) indicates that r_1 is negative and therefore unacceptable as ζ_q. The remaining root r_2 can be written as

$$r_2 = \frac{b - 2cd + \sqrt{(b+2cd)^2 + 8\mu c}}{4|c|}.$$

The quantity inside the square root has magnitude less than $|b+2cd| = -b-2cd$, so that r_2 satisfies

$$r_2 < \frac{b - 2cd - b - 2cd}{4|c|} = -\frac{4cd}{4|c|} = d,$$

and hence is acceptable as ζ_q.

It follows that, when $c \neq 0$, the desired root ζ_q of (23) corresponds to the negative square root in (25).

4.2.2. Two function values, two first derivatives. Here, the four pieces of data that determine $q(\zeta)$ are the values of ϕ and ϕ' at two points, corresponding to $\zeta_1 = 0$ and $\zeta_2 > 0$. The four equations defining the parameters of q are

$$\phi_1 = a - \mu \ln d, \tag{27a}$$

$$\phi_1' = b + \frac{\mu}{d}, \tag{27b}$$

$$\phi_2 = a + b\zeta_2 + c\zeta_2^2 - \mu \ln(d - \zeta_2), \tag{27c}$$

$$\phi_2' = b + 2c\zeta_2 + \frac{\mu}{d - \zeta_2}. \tag{27d}$$

The first two equations provide expressions for a and b in terms of d:

$$a = \phi_1 + \mu \ln d \qquad \text{and} \qquad b = \phi_1' - \frac{\mu}{d}. \tag{28}$$

Substitution for b in (27d) gives an expression for c in terms of d:

$$c = \frac{1}{2\zeta_2}\left(\phi_2' - \phi_1' + \frac{\mu}{d} - \frac{\mu}{d - \zeta_2}\right). \tag{29}$$

We then substitute the formulas for a, b, and c into (27c), re-arrange and obtain the nonlinear equation

$$\ln\left(\frac{d - \zeta_2}{d}\right) + \frac{\zeta_2}{2}\left(\frac{1}{d} + \frac{1}{d - \zeta_2}\right) - k_3 = 0, \tag{30}$$

where

$$k_3 = \frac{1}{\mu}\left(\frac{\zeta_2}{2}(\phi_1' + \phi_2') - (\phi_2 - \phi_1)\right). \tag{31}$$

Introducing the variable $z = 1 - \zeta_2/d$, so that $0 < z < 1$, the equation (30) becomes

$$\psi_3(z) = \ln z + \frac{1}{2}\left(\frac{1}{z} - z\right) - k_3 = 0. \tag{32}$$

Care must be taken in solving for a zero of this equation because of unboundedness in the logarithm and reciprocal functions for z near zero. It is straightforward to show that (i) $\lim_{z \to 0_+} \psi_3(z) = +\infty$ and $\psi_3(1) = -k_3$, and (ii) $\psi_3'(z) < 0$ and $\psi_3''(z) > 0$ for z in $(0, 1)$. We conclude that $\psi_3(z)$ has a unique zero in $(0, 1)$ if $k_3 > 0$. The condition $k_3 > 0$ means that ϕ is locally rising more rapidly than a quadratic, in the sense that the average of ϕ_1' and ϕ_2' exceeds the slope of the straight line joining ϕ_1 and ϕ_2. If $k_3 \leq 0$, $q(\zeta)$ should not be used as an interpolant.

Newton's method is guaranteed to converge monotonically if the initial iterate is chosen so that $\psi_3 > 0$. Except when k_3 is extremely large, this can easily be assured by choosing the starting point from a suitable table. If k_3 is large, z must be small; hence we can simply set $z = \epsilon$ for some suitably small ϵ whenever k_3 is greater than a specified value.

Given z, the values of d, c, and b may be obtained from the relations $d = \zeta_2/(1-z)$, (28), and (29), and then used in a numerically satisfactory form to compute the desired root ζ_q.

Consider example (11), with $q(\zeta)$ fitted to

$$\zeta_1 = 0 \quad \text{and} \quad \zeta_2 = \alpha_b = .02634, \tag{33}$$

for which

$$\phi_1 = -.9822, \quad \phi_1' = -3.292, \quad \phi_2 = -1.056, \quad \text{and} \quad \phi_2' = .5864.$$

Since $\phi_2' > 0$, the minimizer is bracketed in $(0, \zeta_2)$.

The value of k_3 defined by (31) is 3.798. Starting with $z = .07$ (chosen by table lookup), two iterations of Newton's method applied to ψ_3 produce $z = .07832$, with $\psi_3 = 1.486 \times 10^{-7}$. We then obtain $d = .02858$, $c = -4.518$, $b = -3.642$, and $\zeta_q = .02600$. The cubic fitted to the same data has a minimizer at $\zeta = .02507$. The minimizers of $q(\zeta)$ and the cubic are rather similar for this example because the minimizer is bracketed; this feature underlies our suggestion in §5 of using $q(\zeta)$ only when the minimizer is not bracketed.

To illustrate the benefits of using $q(\zeta)$ when the minimizer is not bracketed, consider the same example, but this time with

$$\zeta_2 = \tfrac{1}{2}\alpha_b, \quad \text{so that} \quad \phi_2 = -1.025 \text{ and } \phi_2' = -3.112. \tag{34}$$

The minimizer of $q(\zeta)$ is now $\zeta_q = .02604$ (still a reasonably good estimate of α^*), whereas the minimizer of the interpolating cubic is the infeasible step .05615.

4.2.3. Two function values, one first derivative, one second derivative. The approximating function $q(\zeta)$ is also appropriate when second derivatives of the problem functions are available. In particular, when Newton's method is used to minimize the barrier function, the second derivative of ϕ at $\alpha = 0$ is known. At the first iteration of the line search, the function $q(\zeta)$ can then be chosen to match the data ϕ_1, ϕ_1', ϕ_1'', and ϕ_2.

The defining equations are

$$\begin{aligned} &\phi_1 = a - \mu \ln d, \quad \phi_1' = b + \frac{\mu}{d}, \quad \phi_1'' = 2c + \frac{\mu}{d^2}, \\ &\phi_2 = a + b\zeta_2 + c\zeta_2^2 - \mu \ln(d - \zeta_2). \end{aligned} \tag{35}$$

After the first three equations are arranged to express a, b, and c in terms of d, the fourth equation yields

$$\ln(d - \zeta_2) - \ln d + \frac{\zeta_2}{d} + \frac{\zeta_2^2}{2d^2} = \frac{1}{\mu}(\phi_1 + \zeta_2\phi_1' + \tfrac{1}{2}\zeta_2^2\phi_1'' - \phi_2). \tag{36}$$

If we define $z = 1 - \zeta_2/d$, so that $0 < z < 1$, the relation (36) becomes

$$\psi_4(z) = \ln z + 1 - z + \tfrac{1}{2}(1 - z)^2 + k_4 = 0, \tag{37}$$

where k_4 is given by

$$k_4 = \frac{1}{\mu}(\phi_2 - \phi_1 - \zeta_2\phi_1' - \tfrac{1}{2}\zeta_2^2\phi_1''). \tag{38}$$

Analysis of ψ_4 shows that (i) $\lim_{z \to 0_+} \psi_4(z) = -\infty$ and $\psi_4(1) = k_4$, and (ii) $\psi_4'(z) > 0$ and $\psi_4''(z) < 0$ for all z in $(0, 1)$. It follows that ψ_4 has a unique zero in $(0, 1)$ if $k_4 > 0$. Positivity of k_4 means that the value of ϕ at ζ_2 exceeds that of the

quadratic fitted to ϕ_1, ϕ_1', and ϕ_1'' at ζ_1. If $k_4 \leq 0$, the special interpolant $q(\zeta)$ should be rejected.

Newton's method applied to ψ_4 will converge monotonically if the starting point is such that $\psi_4(z) < 0$. As in §4.2.2, this property can be assured by appropriate selection of a starting point based on the value of k_4 and a small table. To avoid difficulties when z is very close to zero, a suitable cutoff value of k_4 can be chosen. For example, if $k_4 = 10.01$, the solution of $\psi_4(z) = 0$ is $z = 10^{-5}$; hence z can simply be taken as 10^{-5} if $k_4 \geq 10.01$.

After z has been computed, d is given by $\zeta_2/(1-z)$, and b and c are easily obtained from (35). The value of ζ_q is then calculated as described in §4.2.1.

Once again, consider example (11), with ζ_1 and ζ_2 as in (33). Since p is the Newton direction, the second derivative of ϕ at $\zeta = 0$ is simply $\phi''(0) = -\phi'(0)$. The values to which $q(\zeta)$ is fitted are thus

$$\begin{aligned} &\zeta_1 = 0, && \phi_1 = -.9822, \quad \phi_1' = -3.292, \quad \phi_1'' = 3.292, \\ &\zeta_2 = .02634, && \phi_2 = -1.056. \end{aligned}$$

The value of k_4 defined by (38) is 1.197. Starting with $z = .07$ (chosen by table lookup), two iterations of Newton's method applied to ψ_4 produce $z = .07861$, with $\psi_4 = -2.552 \times 10^{-5}$. We then have $d = .02859$, $c = -4.470$, $b = -3.642$, and $\zeta_q = .02601$.

The cubic fitted to the same data has a minimizer at the infeasible step $\zeta = .04011$. In contrast to the example in §4.2.2, the cubic here does not "recognize" that the minimizer is bracketed, since information about the positive derivative of ϕ at ζ_2 is not available.

5. Implementation. The most effective application of the techniques described in this paper varies with the software design criteria of each researcher—in particular, the relative importance attached to features such as simplicity, generality, and reliability. Issues not specifically considered here, such as specialization to convex programs, are also important in certain contexts. We shall therefore not attempt to treat all possible implementation issues, many of which echo those of unconstrained optimization and have been described in detail in the references cited in §2.1. Rather, we briefly indicate the basic ideas needed to incorporate the strategies of this paper into an existing safeguarded polynomial line search that generates a new trial step based on either interpolation or safeguarding.

At each line search iteration, it must be decided whether to compute the interpolated point using the special functions of §4 or standard polynomials. At the first iteration, the principal guideline is to use the special functions if $\alpha^{(0)}$ is taken as α_b (see §3). After the first iteration, the special functions are tried if the minimizer does not appear to be bracketed, or if a constraint violation occurred at the most recent trial step. Each special function is meaningful only if a certain constant depending on the given data is positive; if it is not, a standard polynomial interpolant is constructed.

The following block of pseudo-code sketches a line search procedure based on the existing routines in the NAG library [NAG]. Two guiding motivations are: (1) the special functions tend to be most useful in limiting the step during the early iterations of a given line search; and (2) when the minimizer is bracketed within a strictly feasible interval of uncertainty, standard low-order polynomials are used unless infeasibility occurs. The specific criteria used to update the interval of uncertainty and to decide when the desired step is bracketed are described in detail in [GM74]; a general discussion of these issues may be found in [BR73].

Pseudo-Code for Barrier Line Search

1. Initialize the interval of uncertainty; its left endpoint α_{lo} is set to zero. Set α_{best} to zero.
2. Calculate $\alpha^{(0)}$ using the procedure of §3.2. If $\alpha^{(0)} = \alpha_b$, set the logical value 'barrier' to 'true'; otherwise, set 'barrier' to 'false'. Set α_{trial} to $\alpha^{(0)}$.
3. Evaluate the constraints at α_{trial}. If all constraints are strictly satisfied, evaluate f and proceed to step 5. Otherwise, go on to step 4.
4. Let $\mathcal{V}$ denote the set of indices of violated constraints at α_{trial}; set 'barrier' to 'true'. For each index $j \in \mathcal{V}$, calculate the secant step σ_j to the zero of the linearized constraint j, fitting a straight line between the constraint values at α_{lo} and at α_{trial}. Let $\bar{\alpha}$ be the smallest value of $\{\sigma_j\}$ for $j \in \mathcal{V}$, with $\bar{\jmath}$ the corresponding index. The new value of α_{trial} is the value of α_b calculated as in §3.2, defined with respect to the current interval of uncertainty. Go back to step 3.
5. Update α_{best} and the interval of uncertainty $[\alpha_{\text{lo}}, \alpha_{\text{hi}}]$. See [GM74] for details of these procedures. If the minimizer is bracketed, set 'barrier' to 'false'.
6. If the sufficient decrease criteria are satisfied at α_{best} or if the interval of uncertainty is sufficiently small, exit.
7. If 'barrier' is 'false', go to step 8. If 'barrier' is 'true', calculate the appropriate quantity k_i, depending on the form of the special interpolant (see §4). If $k_i > 0$, calculate an increment ζ to α_{lo} for the chosen special function as described in §4, and go to step 9. If $k_i \leq 0$, set 'barrier' to 'false' and go on to step 8.
8. Calculate an increment ζ using polynomial interpolation.
9. Given the increment ζ, apply a standard safeguarding strategy. The new trial step α_{trial} is then either $\alpha_{\text{lo}} + \zeta$ or a safeguarded step. Go back to step 3.

The significance of the line search in the overall computational effort needed to solve a given problem obviously depends on the problem, the starting point, and the stringency of the sufficient decrease requirements. When relaxed acceptance criteria apply, simply choosing $\alpha^{(0)}$ as described in §3.2 very often leads to a satisfactory reduction in Φ, so that no line search iterations are required. For example, in the calculations of §3.2 for (11), the point α_b produces a large reduction in the barrier function and a positive value of ϕ'; hence the sufficient decrease conditions (5) are satisfied at α_b and no further points need to be computed. This situation is analogous to that in linear programming, where a sufficient decrease is almost always attained at the initial trial step.

We have performed computational tests of Newton and quasi-Newton methods applied to minimize the sequence of barrier functions (2) associated with approximately 40 constrained test problems, mostly with nonlinear constraints, with a variety of starting points for each problem. Use of the procedures described here led to an overall average reduction of around 20% in the number of constraint evaluations compared to using a standard safeguarded procedure modified so that backtracking occurs when an infeasible step is taken.

Other researchers have also found these procedures to be helpful, even when all the constraints are linear. Nash and Sofer [NS93] report that, when applying a barrier method to large-scale bound-constrained problems, use of our specialized interpolating functions led to a 27% overall reduction in gradient evaluations compared to the standard line search in their general truncated Newton software package.

The techniques described in this paper thus provide an array of possibilities that

might be helpful in special line searches for barrier functions.

Acknowledgment. We are grateful for helpful comments and suggestions from the referees.

REFERENCES

[AN90] K. M. ANSTREICHER, *On long step path following and SUMT for linear and quadratic programming*, Report, George Washington University, Washington, DC, 1990.

[ADRT] K. M. ANSTREICHER, D. DEN HERTOG, C. ROOS, AND T. TERLAKY, *A long step barrier method for convex quadratic programming*, Algorithmica 10 (1993), 365-382.

[BR73] R. P. BRENT, *Algorithms for Minimization without Derivatives*, Prentice-Hall, Englewood Cliffs, New Jersey, 1973.

[BN89] R. H. BYRD AND J. NOCEDAL, *A tool for the analysis of quasi-Newton methods with application to unconstrained optimization*, SIAM J. Numer. Anal., 26 (1989), pp. 727–739.

[DRT] D. DEN HERTOG, C. ROOS, AND T. TERLAKY, *On the classical logarithmic barrier function method for a class of smooth convex programming problems*, Report 90-28, Delft University of Technology, Delft, the Netherlands, April 1990.

[DM77] J. E. DENNIS, JR. AND J. J. MORÉ, *Quasi-Newton methods, motivation and theory*, SIAM Review, 19 (1977), pp. 46–89.

[DS83] J. E. DENNIS, JR. AND R. B. SCHNABEL, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, Englewood Cliffs, New Jersey, 1983.

[DS89] ———, *A view of unconstrained optimization*, in Optimization, G. L. Nemhauser, A. H. G. Rinnooy Kan, and M. J. Todd, eds., North-Holland, New York, 1989, pp. 1–72.

[FI79] A. V. FIACCO, *Barrier methods for nonlinear programming*, in Operations Research Support Methodology, A. G. Holzman, ed., Marcel Dekker, New York, 1979.

[FM68] A. V. FIACCO AND G. P. MCCORMICK, *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*, John Wiley and Sons, New York, 1968. Republished by SIAM, Philadelphia, 1990.

[FL87] R. FLETCHER, *Practical Methods of Optimization*, 2nd ed., John Wiley and Sons, New York, 1987.

[FLMC] R. FLETCHER AND A. P. MCCANN, *Acceleration techniques for nonlinear programming*, in Optimization, R. Fletcher, ed., Academic Press, New York, 1969, pp. 203–213.

[GM74] P. E. GILL AND W. MURRAY, *Safeguarded steplength algorithms for optimization using descent methods*, Report NAC 37, National Physical Laboratory, Teddington, England, 1974.

[GMW] P. E. GILL, W. MURRAY, AND M. H. WRIGHT, *Practical Optimization*, Academic Press, London, 1981.

[GMSW] P. E. GILL, W. MURRAY, M. A. SAUNDERS, AND M. H. WRIGHT, *Constrained nonlinear programming*, in Optimization, G. L. Nemhauser, A. H. G. Rinnooy Kan, and M. J. Todd, eds., North-Holland, New York, 1989, pp. 171–210.

[GT89] D. GOLDFARB AND M. J. TODD, *Recent developments in linear programming*, in Optimization, G. L. Nemhauser, A. H. G. Rinnooy Kan, and M. J. Todd, eds., North-Holland, New York, 1989, pp. 73–170.

[GO92] C. C. GONZAGA, *Path-following methods for linear programming*, SIAM Review, 34 (1992), pp. 167–224.

[JA90] F. JARRE, *Interior-point methods for convex programming*, Report SOL 90-16, Department of Operations Research, Stanford University, Stanford, California, 1990.

[JA92] F. JARRE, *An efficient line-search for logarithmic barrier functions*, Technical Report 188, Institüt für Angewandte Mathematik, University of Würzburg, Würzburg, Germany, 1992.

[JS91] F. JARRE AND M. A. SAUNDERS, *Practical aspects of an interior-point method for convex programming*, Report SOL 91-9, Department of Operations Research, Stanford University, Stanford, California, 1991.

[KA84] N. K. KARMARKAR, *A new polynomial-time algorithm for linear programming*, Combinatorica, 4 (1984), pp. 373–395.

[LFR] L. S. LASDON, R. L. FOX, AND M. W. RATNER, *An efficient one-dimensional search procedure for barrier functions*, Math. Programming, 4 (1973), pp. 279–295.

[MS90] S. MEHROTRA AND J. SUN, *An interior point algorithm for solving smooth convex programs based on Newton's method*, in Mathematical Developments Arising from Linear Programming, J. C. Lagarias and M. J. Todd, eds., Contemporary Mathematics, Volume 114, American Mathematical Society, Providence, Rhode Island, 1990, pp. 265–294.

[MW76] W. MURRAY AND M. H. WRIGHT, *Efficient linear search algorithms for the logarithmic barrier function*, Report SOL 76–18, Department of Operations Research, Stanford University, Stanford, California, 1976.

[NAG] *NAG Fortran Library Manual, Mark* 15, Numerical Algorithms Group, Ltd., Downers Grove, Illinois, 1991.

[NS93] S. G. NASH AND A. SOFER, *A barrier method for large-scale constrained optimization*, ORSA Journal on Computing, 5 (1993), pp. 40–53.

[NN89] J. E. NESTEROV AND A. S. NEMIROVSKY, *Self-concordant Functions and Polynomial-time Methods in Convex Programming*, Central Economic and Mathematical Institute, USSR Academy of Science, Moscow, USSR, 1989.

[OR70] J. M. ORTEGA AND W. C. RHEINBOLDT, *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, New York, 1970.

[RI91] U. T. RINGERTZ, *Optimal design of nonlinear shell structures*, Report FFA TN 91-18, Aeronautical Research Institute of Sweden, Bromma, Sweden, 1991.

[WR76] M. H. WRIGHT, *Numerical Methods for Nonlinearly Constrained Optimization*, Ph. D. thesis, Computer Science Department, Stanford University, Stanford, California, 1976.

[WR92] ———, *Interior methods for constrained optimization*, in Acta Numerica 1992, A. Iserles, ed., Cambridge University Press, New York, 1992, pp. 341–407.

SIAM J. OPTIMIZATION
Vol. 4, No. 2, pp. 247–261, May 1994
© 1994 Society for Industrial and Applied Mathematics
002

SUPERLINEARLY CONVERGENT $O(\sqrt{n}L)$-ITERATION INTERIOR-POINT ALGORITHMS FOR LINEAR PROGRAMMING AND THE MONOTONE LINEAR COMPLEMENTARITY PROBLEM*

KEVIN MCSHANE†

Abstract. Zhang and Tapia have recently developed an interior-point algorithm for the linear programming problem that has the property that the duality gap converges to zero q-quadratically in the nondegenerate case and (if the iterates converge) q-superlinearly in the degenerate case. Their algorithm also solves the integer model for LP in $O(nL)$ iterations, where L is the input length of the LP. This paper presents an algorithm which maintains the quadratic (superlinear) convergence property but solves LP in $O(\sqrt{n}L)$ iterations in the integer model. Also, the algorithm is generalized into an algorithm which solves LCP. The LCP algorithm (which is similar to an adaptive path-following algorithm proposed by Mizuno, Yoshise, and Kikuchi) has similar local convergence properties if we assume that the iterates converge to a strictly complementary solution.

Key words. linear programming, linear complementarity problem, primal-dual interior-point algorithms, superlinear convergence, quadratic convergence, polynomiality

AMS subject classifications. 90C05, 65K05

1. Introduction.

This paper presents interior-point algorithms for the linear programming problem (LP) and the monotone linear complementarity problem (LCP) with the following two properties:

(1) In both the LP and LCP cases, the duality gap converges to zero at least q-linearly. For LP, the convergence is q-quadratic in the nondegenerate case and, if degeneracy exists and the primal and dual iterates converge to optimal solutions, the duality gap converges to zero q-superlinearly. In the LCP case, the same local convergence results apply, except that we must additionally assume that the iterates converge to a strictly complementary optimal solution.

(2) Under the integer model, the algorithms require at most $O(\sqrt{n}L)$ iterations, where L is the encoding length of the problem.

One of the first papers to consider the question of local convergence of the duality gap in interior point LP algorithms was Iri and Imai [7]. The paper presented an algorithm that, with an exact line search procedure, exhibits q-quadratic convergence if the LP is nondegenerate. Tsuchiya and Tanabe [26] followed by showing q-quadratic convergence for the Iri and Imai algorithm under a weaker nondegeneracy assumption. Yamashita [27] then showed that a variant of the Iri and Imai algorithm not only converges q-quadratically (assuming nondegeneracy), but also could be used to solve LP in $O(nL)$ iterations if the data is integer or rational.

Since then there has been a flurry of activity in the development of algorithms with improved local convergence properties. Coleman and Li [3] developed a quadratically convergent algorithm for LP problems with bounded variables. In [33], Zhang and Tapia present a primal-dual algorithm that solves the integer model for LP in $O(nL)$ iterations and converges q-superlinearly if the iterates converge to an optimal solution and q-quadratically under a nondegeneracy assumption. The LP algorithm developed below has the same local convergence properties as the Zhang and Tapia [33] algorithm, but solves the integer model for LP in $O(\sqrt{n}L)$ iterations. This is currently the best known iteration complexity bound for LP, first obtained by Renegar [22]. Many algorithms subsequently achieved this bound, including the path-following algorithms of Gonzaga [5], Kojima, Mizuno, and Yoshise [11], and Monteiro

* Received by the editors November 4, 1991; accepted for publication (in revised form) February 26, 1993.
† 2537 Villanova Drive, Vienna, Virginia 22180.

and Adler [21], and the potential reduction algorithms of Ye [29], Freund [4], Kojima, Mizuno, and Yoshise [12], and Todd and Ye [24].

After the initial version of this paper appeared, we discovered that Ye, Tapia, and Zhang [32] had (simultaneously) shown that the predictor-corrector algorithm of Mizuno, Todd, and Ye [19] has the same local and global convergence properties as the algorithm presented below. In fact, Ye, Güler, Tapia, and Zhang [31] and Mehrotra [16] later demonstrated q-quadratic convergence of the duality gap to zero for the predictor-corrector algorithm in the degenerate case without the assumption of convergence of the iterates (a similar result was shown for the Iri and Imai algorithm in Tsuchiya [25]). The predictor-corrector algorithm, however, requires the formation and factorization of two matrices each iteration while the algorithm developed below requires only one. An interesting related paper is Mehrotra [15].

Interest in similar algorithms for LCP was apparent in [35], where Zhang, Tapia, and Potra discussed sufficient conditions for superlinear and polynomial-time convergence, and in [10], where Kojima, Megiddo, and Noma develop an interior point algorithm for LCP for which the duality gap converges to zero q-quadratically assuming a unique, strictly complementary solution exists. Later, Ji, Potra, Tapia, and Zhang [9] proposed an algorithm with q-superlinear convergence assuming the algorithm converges to a strictly complementary solution, but which also solves the integer model for LCP in $O(nL)$ iterations. Subsequently, Ji, Potra, and Huang [8] extended the predictor-corrector algorithm to the LCP case and showed that it is an $O(\sqrt{n}L)$ iteration algorithm with similar local convergence properties, but which also converges q-quadratically if there is a unique strictly complementary solution. The LCP algorithm developed below has these same convergence properties. Recently, Ye and Anstreicher [30] showed that if the LCP has a strictly complementary solution, the duality gap in the predictor-corrector algorithm converges to zero q-quadratically in the degenerate case even if the iterates are not assumed to converge. As in the LP situation, the predictor-corrector algorithm for LCP requires two factorizations per iteration while the LCP algorithm below requires only one.

The remainder of this paper is organized as follows. In §2, a standard form for the LP and LCP problems is described and approximate centering is discussed. The proposed LP and LCP algorithms are discussed in §3. Section 4, the main section, presents the convergence results, and §5 contains some concluding remarks.

2. Preliminaries. In order to highlight the similarities between the LP and LCP algorithms developed below, we will describe them in parallel. We shall see that very little extra work is required to extend LP results to the LCP case. In this section we will standardize the formulations of the problems, and discuss "approximate centering."

2.1. The LP and LCP problems in a standard form. Consider the primal-dual pair of linear programs:

$$\text{(P)} \qquad \begin{aligned} &\min\ c'x \\ \text{s.t.}\quad &Ax = b, \\ &x \geq 0, \end{aligned}$$

$$\text{(D)} \qquad \begin{aligned} &\max\ b'y, \\ \text{s.t.}\quad &A'y + s = c, \\ &s \geq 0, \end{aligned}$$

which $A \in R^{m\times n}$, c and x are n-vectors, and b is an m-vector. A is assumed to have full row rank. Todd and Ye [24] showed that the primal and dual linear programming problems can be

reformulated into the following primal-dual form:

$$\text{(LP)} \qquad \text{s.t.} \quad \begin{array}{l} \min x's, \\ Ax = b, \\ \hat{P}s = \hat{P}c, \\ x,\ s \geq 0, \end{array}$$

where $\hat{P}$ projects orthogonally onto the null space of A. The matrix $\hat{P}$ is used only for ease of exposition—the LP algorithm we develop below does not require that the matrix $\hat{P}$ be formed. Given feasible points x, y, and s, the quantity $x's$ is an alternative expression for the duality gap $c'x - b'y$.

The monotone linear complementarity problem can be formulated similarly as

$$\text{(LCP)} \qquad \text{s.t.} \quad \begin{array}{l} \min x's, \\ Mx + q = s, \\ x,\ s \geq 0, \end{array}$$

where $M \in R^{n\times n}$ is positive semidefinite (not necessarily symmetric), and q, s, and x are n-vectors. As in the LP case, the optimal objective function value of LCP is zero.

We will use the following notation. Given strictly feasible x_0 and s_0 (i.e., vectors that satisfy the constraints in LP or LCP and have strictly positive components), feasible directions dx and ds (i.e., $A\,dx = 0$ and $\hat{P}\,ds = 0$ in the LP case or $M\,dx = ds$ in the LCP case), and a positive step length λ, we call x_0 and s_0 *strictly complementary* if for each component i, $x_0^i s_0^i = 0$ and $x_0^i + s_0^i > 0$. Define

$$\begin{array}{rlrlrl} x(\lambda) &:= x_0 + \lambda\,dx, & s(\lambda) &:= s_0 + \lambda\,ds, & & \\ X_0 &:= \operatorname{diag}(x_0), & S_0 &:= \operatorname{diag}(s_0), & & \\ D_x &:= \operatorname{diag}(dx), & D_s &:= \operatorname{diag}(ds), & D_0 &:= (X_0 S_0^{-1})^{1/2}, \\ X(\lambda) &:= X_0 + \lambda\,D_x, & S(\lambda) &:= S_0 + \lambda\,D_s, & & \\ \Delta_0 &:= \dfrac{x_0' s_0}{n}, \quad \text{and} & \Delta(\lambda) &:= \dfrac{x(\lambda)'s(\lambda)}{n}. & & \end{array}$$

If a subscript k appears, it refers to the corresponding quantity for the kth iteration of the algorithm. All norms below are 2-norms unless otherwise denoted by a subscript. We will refer to the LP algorithm derived below as Algorithm I and we will call the LCP algorithm Algorithm II.

2.2. Approximate centering. Some interior-point LP and LCP algorithms can behave badly near the boundary of the feasible region (Megiddo and Shub [14]). To overcome this problem, algorithms were devised that steer clear of the boundary by forcing the iterates to stay close to the *central path*. Define $\Delta := x's/n$ and e as the n-vector of ones. The central path is defined as

$$\Gamma := \{(x, s) \mid x \text{ and } s \text{ are strictly feasible}, XSe = \Delta e\},$$

and is the subject of a large amount of research (see, for example, [1], [2], and [13]).

One particularly successful method used to avoid the boundary requires that all iterates obey the *approximately centered condition*

$$\|XSe - \Delta e\| \leq \beta\,\Delta, \tag{1}$$

where $\beta \in [0, 1)$ is a fixed parameter. This technique was first used by Monteiro and Adler [21] and for LCP by Kojima, Mizuno, and Yoshise [11], and later by many others. We will

enforce this condition for all iterates of Algorithms I and II, that is, Algorithms I and II are *path-following algorithms*. The approximately centered condition appears to play an important role in developing algorithms with improved local convergence properties. The condition allows a sufficient decrease in the objective value to be maintained from iteration to iteration.

3. Motivation for the algorithms. In this section we will present the search directions we will use and formally state the algorithms. We will motivate the search directions by examining the directions used by certain potential reduction algorithms, although we could have motivated them by using the generic path-following framework that appears in [33].

3.1. The search directions. The search directions we will use are similar to those used by the potential reduction algorithms of Todd and Ye [24] and Kojima, Mizuno, and Yoshise [12], although Algorithms I and II are not potential reduction algorithms. Rather than attempting to reduce the objective value $x's$ of LP or LCP each iteration, the algorithms of Todd and Ye [24] and Kojima, Mizuno, and Yoshise [12] reduce each iteration the value of the potential function

$$\phi(x, s) := \rho \log(x's) - \sum_i \log(x^i s^i),$$

where ρ is a fixed parameter greater than n, typically chosen as $n + \sqrt{n}$. It can be shown that the objective function value for LP or LCP, $x's$, can be made arbitrarily small by finding strictly feasible x and s with negative enough potential value (see [12]).

Given strictly feasible points x_0 and s_0 for LP or LCP, define e as the n-vector of ones and

$$v_0 := \frac{n\Delta_0}{\rho}(X_0S_0)^{-1/2}e - (X_0S_0)^{1/2}e. \tag{2}$$

In the LP case, Todd and Ye [24] proposed the following projected scaled steepest descent directions for the potential function ϕ:

$$\begin{bmatrix} dx \\ ds \end{bmatrix} = \begin{bmatrix} DPv_0 \\ D^{-1}(I - P)v_0 \end{bmatrix}, \tag{3}$$

where P projects orthogonally onto the null space of AD.

The LCP algorithm of Kojima et al. uses directions that are the unique solution to the following system of equations:

$$\begin{bmatrix} M & -I \\ D_0^{-1} & D_0 \end{bmatrix} \begin{bmatrix} dx \\ ds \end{bmatrix} = \begin{bmatrix} 0 \\ v_0 \end{bmatrix}. \tag{4}$$

Equivalently, one could solve the system

$$(M + D_0^{-2})\, dx = D_0^{-1}v_0$$

for dx and then form ds via $M\, dx = ds$.

It has been known for some time that the search directions used by many of the theoretically efficient interior point algorithms are linear combinations of "centering" and "affine scaling" directions (see, for example, Mitchell and Todd [17]). The relative weights on the directions (parametrized by ρ above) are fixed for many of the existing polynomial-time algorithms, but authors have suggested "adaptive path following" strategies (Ye [28], Mizuno, Yoshise, and Kikuchi [20]; see also Mizuno and Todd [18], Mizuno, Todd, and Ye [19]) that allow the weights to change from iteration to iteration while maintaining polynomial-time convergence.

Algorithms I and II are based on the adaptive path-following approach and are quite similar to an LCP algorithm proposed in [20]. We will usually index ρ with a subscript which indicates iteration number.

We now describe the search directions we will use. The vector v_0 in (2) can be rewritten as

$$\begin{aligned} v_0 &:= -\frac{n}{\rho_0}(X_0S_0)^{-1/2}(X_0S_0e - \Delta_0 e) - \frac{(\rho_0 - n)}{\rho_0}(X_0S_0)^{1/2}e, \\ &:= \frac{n}{\rho_0}\bar{v}_0 + \frac{(\rho_0 - n)}{\rho_0}\tilde{v}_0, \end{aligned} \tag{5}$$

where

$$\bar{v}_0 := -(X_0S_0)^{-1/2}(X_0S_0e - \Delta_0 e) \quad \text{and} \quad \tilde{v}_0 := -(X_0S_0)^{1/2}e.$$

For convenience, we will usually suppress the iteration index subscript for the search directions. In the LP case, define

$$\overline{dx} := D_0 P \bar{v}_0, \tag{6a}$$

$$\overline{ds} := D_0^{-1}(I - P)\bar{v}_0, \tag{6b}$$

$$\tilde{dx} := D_0 P \tilde{v}_0, \quad \text{and} \tag{7a}$$

$$\tilde{ds} := D_0^{-1}(I - P)\tilde{v}_0. \tag{7b}$$

For LCP, define $\overline{dx}$ and $\overline{ds}$ as the unique solution to

$$\begin{bmatrix} M & -I \\ D_0^{-1} & D_0 \end{bmatrix} \begin{bmatrix} \overline{dx} \\ \overline{ds} \end{bmatrix} = \begin{bmatrix} 0 \\ \bar{v}_0 \end{bmatrix} \quad \text{and} \tag{8}$$

$\tilde{dx}$ and $\tilde{ds}$ as the solution to

$$\begin{bmatrix} M & -I \\ D_0^{-1} & D_0 \end{bmatrix} \begin{bmatrix} \tilde{dx} \\ \tilde{ds} \end{bmatrix} = \begin{bmatrix} 0 \\ \tilde{v}_0 \end{bmatrix}. \tag{9}$$

Consequently, for both LP and LCP we have

$$dx = \frac{n}{\rho_0}\overline{dx} + \frac{\rho_0 - n}{\rho_0}\tilde{dx} \quad \text{and} \quad ds = \frac{n}{\rho_0}\overline{ds} + \frac{\rho_0 - n}{\rho_0}\tilde{ds}. \tag{10}$$

3.2. The algorithms. There is a basic relationship between λ, ρ_0, and the approximately centered condition. Given initial points x_0 and s_0 to LP or LCP that are approximately centered with parameter β, we will later show that the updated points $x(\lambda)$ and $s(\lambda)$ will also be approximately centered with parameter β and strictly feasible if the following relationship between λ and ρ_0 holds:

$$0 < \lambda \leq \min\left\{\frac{n\beta\Delta_0}{\rho_0\|D_xD_se\|}, 1\right\}. \tag{11}$$

After $\overline{dx}$, $\overline{ds}$, $\tilde{dx}$, and $\tilde{ds}$ have been computed, Algorithms I and II attempt to choose a value for ρ that makes the leftmost expression in braces above equal to one, that is, choose ρ as the largest root of the fourth degree polynomial

$$g(\rho) := \rho^4 \|D_x D_s e\|^2 - \rho^2 n^2 \beta^2 \Delta_0^2. \tag{12}$$

Note from (10) that $\|D_x D_s e\|$ is a function of ρ.

More formally:

ALGORITHMS I AND II

Step (0) Given strictly feasible x_0 and s_0 (for LP or LCP) which satisfy (1) for some fixed $\beta \leq 0.5$, set $k := 0$.

Step (1) Suppose we are given strictly feasible x_k and s_k (for LP or LCP) that are approximately centered with parameter β. Compute $\tilde{dx}$ and $\tilde{ds}$ from (7) or (9), and, using a ratio test, check to see if using the maximal feasible step length solves the problem. If not, compute $\overline{dx}$ and $\overline{ds}$ from (6) or (8) and choose ρ_k as the largest root of the function g in (12). Compute the search directions from (10).

Step (2) Update $x_{k+1} := x_k + dx$ and $s_{k+1} := s_k + ds$, and if the termination criteria are not satisfied, set $k := k + 1$ and go to Step (1).

4. Convergence results. In this section we will show that Algorithms I and II are well defined, and if λ and ρ are chosen so that condition (11) is satisfied, the updated iterates will be strictly feasible and approximately centered. This section also contains our local and global convergence results.

4.1. Basic properties and polynomial complexity.

PROPOSITION 4.1. *Suppose we are given strictly feasible points x_0 and s_0 for* LP *or* LCP, *the feasible directions dx and ds from* (10), *a value for $\rho_0 > n$, and a step length $\lambda > 0$. Then*

$$\text{(a)} \quad x(\lambda)'s(\lambda) = \left(1 - \lambda \frac{(\rho_0 - n)}{\rho_0}\right) x_0' s_0 + \lambda^2 (dx)'(ds); \tag{13}$$

$$\begin{aligned} \text{(b)} \quad X(\lambda)S(\lambda)e - \Delta(\lambda)e &= (1-\lambda)(X_0 S_0 e - \Delta_0 e) \\ &\quad + \lambda^2 \left(D_x D_s e - \frac{(dx)'(ds)}{n} e\right). \end{aligned} \tag{14}$$

Proof. The proof is straightforward, hence it is omitted. □

PROPOSITION 4.2. *Suppose (x_0, s_0) is an approximately centered strictly feasible pair with parameter β. For any $p > n$, the directions* (10) *satisfy $(dx)'(ds) \geq 0$ and*

$$\|D_x D_s e\| \leq \frac{\|v_0\|^2}{2} \leq \frac{n\Delta_0((\rho - n)^2 + n\beta^2/(1-\beta))}{2\rho^2}. \tag{15}$$

Proof. In the LP case, it can be easily shown that $(dx)'(ds) = 0$. In the LCP case, from (4), $M\,dx = ds$, therefore $dx' M\,dx = (dx)'(ds)$. Since M is positive semidefinite, $(dx)'(ds)$ is nonnegative. Now, observe that

$$\begin{aligned} \|D_x D_s e\|^2 &= \|D_x D_0^{-1} D_0 D_s e\|^2 = (e' D_x^2 D_0^{-2})(D_0^2 D_s^2 e) \\ &\leq \|D_0^{-2} D_x^2 e\|_2 \|D_0^2 D_s^2 e\|_2 \leq \|D_0^{-2} D_x^2 e\|_1 \|D_0^2 D_s^2 e\|_1 = \|D_0^{-1} D_x e\|_2^2 \|D_0 D_s e\|_2^2 \\ &\leq \tfrac{1}{4}(\|D_0^{-1}\,dx\|_2^2 + \|D_0\,ds\|_2^2)^2 \end{aligned}$$

by the arithmetic-geometric mean inequality. In both the LP and LCP cases, $D_0^{-1}\,dx + D_0\,ds = v_0$; therefore

$$\|D_0^{-1}\,dx + D_0\,ds\|^2 = \|v_0\|^2.$$

If we expand the left-hand side and rearrange,

$$\|D_0^{-1}\,dx\|^2 + \|D_0\,ds\|^2 = \|v_0\|^2 - 2(dx)'(ds) \leq \|v_0\|^2, \tag{16}$$

which establishes the first inequality in (15).

Finally, from (5) (note that $\tilde{v}_0$ and $\bar{v}_0$ are orthogonal),

$$\|v_0\|^2 \leq \frac{n^2}{\rho^2}\|(X_0S_0)^{-1/2}\|^2\|X_0S_0e - \Delta_0 e\|^2 + \frac{(\rho - n)^2}{\rho^2}\|(X_0S_0)^{1/2}e\|^2.$$

Approximate centering implies that for each component i, $(1-\beta)\Delta_0 \leq x_0^i s_0^i \leq (1+\beta)\Delta_0$; therefore

$$\|v_0\|^2 \leq \frac{n^2\beta^2\Delta_0}{(1-\beta)\rho^2} + \frac{(\rho - n)^2 n\Delta_0}{\rho^2},$$

from which the final inequality immediately follows. □

THEOREM 4.1. *Suppose that (x_0, s_0) is an approximately centered strictly feasible pair with parameter β and that we are given a value for $\rho > n$. Suppose also that dx and ds be computed from* (10). *Define*

$$\lambda^*(\rho) := \min\left\{\frac{n\beta\Delta_0}{\rho\|D_xD_se\|}, 1\right\},$$

where the fraction within the braces is interpreted as infinity if the denominator is zero. Then any step length $0 < \lambda < \lambda^*$ yields new strictly feasible points that are approximately centered with parameter β, and a step of length λ^* yields either an optimal solution or strictly feasible, approximately centered points.

Proof. For simplicity of notation, denote $\lambda^*(\rho)$ by λ^*, but keep in mind the dependence on ρ. Note also that the value of $\|D_xD_se\|$ depends on ρ. We distinguish two cases: $\|D_xD_se\| = 0$ and $\|D_xD_se\| > 0$.

If $\|D_xD_se\| = 0$ (which implies that $dx'ds = 0$), (14) indicates that

$$\|X(\lambda)S(\lambda)e - \Delta(\lambda)e\| \leq (1-\lambda)\beta\Delta_0$$

for any $\lambda \in (0, 1]$. But from (13),

$$\Delta_0 = \frac{1}{1 - \lambda\dfrac{(\rho - n)}{\rho}}\Delta(\lambda);$$

hence $\Delta(\lambda) > 0$ and

$$\|X(\lambda)S(\lambda)e - \Delta(\lambda)e\| \leq \frac{1-\lambda}{1 - \lambda\dfrac{(\rho - n)}{\rho}}\beta\Delta(\lambda) < \beta\Delta(\lambda)$$

for all $\lambda \in (0, 1]$, which implies that approximate centering is strictly maintained for all $\lambda \in (0, 1]$. This means that for each component i, $(1-\beta)\Delta(\lambda) < x^i(\lambda)s^i(\lambda) < (1+\beta)\Delta(\lambda)$

throughout the interval; therefore $x(\lambda)$ and $s(\lambda)$ have strictly positive components for any $\lambda \in (0, 1]$, in particular for λ^*.

If $\|D_x D_s e\| > 0$, note that

$$\left\| D_x D_s e - \frac{(dx)'(ds)}{n} e \right\|^2 = \|D_x D_s e\|^2 - \frac{1}{n}((dx)'(ds))^2;$$

therefore

$$\left\| D_x D_s e - \frac{(dx)'(ds)}{n} e \right\| \leq \|D_x D_s e\|.$$

Expression (14) implies that for any λ in the interval [0,1]

$$\begin{aligned} \|X(\lambda)S(\lambda)e - \Delta(\lambda)e\| &\leq (1-\lambda)\beta\Delta_0 + \lambda^2 \left\| D_x D_s e - \frac{(dx)'(ds)}{n} e \right\| \\ &\leq (1-\lambda)\beta\Delta_0 + \lambda^2 \|D_x D_s e\|. \end{aligned}$$

But (13) implies

$$\Delta(\lambda) = \left(1 - \lambda\frac{(\rho - n)}{\rho}\right)\Delta_0 + \lambda^2 \frac{(dx)'(ds)}{n},$$

and if we solve (assuming $\lambda \in (0, 1]$) the inequality

$$(1-\lambda)\beta\Delta_0 + \lambda^2 \|D_x D_s e\| < \beta\Delta(\lambda)$$

for λ, we find that approximate centering is maintained for all values of λ such that

$$0 < \lambda < \min\left\{ \frac{n\beta\Delta_0}{\rho\left(\|D_x D_s e\| - \beta\dfrac{(dx)'(ds)}{n}\right)}, 1 \right\}.$$

Note that $(dx)'(ds) \leq \|D_x D_s e\|_1 \leq \sqrt{n}\|D_x D_s e\|_2$, so the denominator above is strictly positive. Also, for all λ such that $0 < \lambda < \lambda^*$, the updated iterates will strictly satisfy (1), and

$$(1-\beta)\Delta(\lambda) < x^i(\lambda)s^i(\lambda) < (1+\beta)\Delta(\lambda)$$

for each component i. Consequently, $x(\lambda)$ and $s(\lambda)$ have either strictly positive components for all $\lambda \in (0, \lambda^*]$, or else $\Delta(\lambda^*) = 0$, which implies that $x(\lambda^*)$ and $s(\lambda^*)$ are optimal solutions. □

COROLLARY 4.1. *Suppose (x_0, s_0) is an approximately centered strictly feasible pair with parameter $\beta < \frac{2}{3}$, and suppose dx and ds are computed from* (10). *Then*

$$g(n + \beta + \sqrt{\alpha n + \beta^2}) \leq 0,$$

where $\alpha := 2\beta - \beta^2/(1-\beta)$, and g is as defined in (12).

Proof. From (15), for any $\rho > n$,

$$\|D_x D_s e\| \leq \frac{n\Delta_0((\rho - n)^2 + n\beta^2/(1-\beta))}{2\rho^2},$$

so $g(\rho) \leq 0$ if $\rho > n$ is chosen so that

$$\frac{n^2\beta^2\Delta_0}{2(1-\beta)\rho^2} + \frac{(\rho-n)^2 n\Delta_0}{2\rho^2} \leq \frac{n\beta\Delta_0}{\rho},$$

or equivalently,

$$\rho^2 - 2(n+\beta)\rho + n^2 + n\beta^2/(1-\beta) \leq 0. \tag{17}$$

By solving (17) for ρ when the inequality sign is replaced with an equality we find that the largest root of the resulting quadratic equation is

$$\rho_* = n + \beta + \sqrt{\left(2\beta - \frac{\beta^2}{1-\beta}\right)n + \beta^2}, \tag{18}$$

which is well defined for $\beta \leq \frac{2}{3}$. If $\beta < \frac{2}{3}$, the coefficient of n under the radical sign is positive. □

THEOREM 4.2. *If during any iteration of Algorithm* I *or Algorithm* II *the function g does not have a root at least as large as* ρ_* *as defined in* (18), *the algorithm will terminate at optimality that iteration.*

Proof. Let $\overline{dx}$, $\overline{ds}$, $\tilde{dx}$, and $\tilde{ds}$ be as defined in (6) and (7) for LP or as (8) and (9) for LCP. Consider the coefficient of ρ^4 in g, which (using (10)) is $\|\tilde{D}_x\tilde{D}_s e\|^2$, where $\tilde{D}_x$ is diag($\tilde{dx}$) and $\tilde{D}_s$ is diag($\tilde{ds}$). If this coefficient is positive, then $g \to \infty$ as $\rho \to \infty$, so by continuity and Corollary 4.1, g must have a root at least as large as ρ_*.

If $\|\tilde{D}_x\tilde{D}_s e\| = 0$, then $(\tilde{dx})'(\tilde{ds}) = 0$. This means that (using (10))

$$(dx)'(ds) = \frac{n^2(\overline{dx})'(\overline{ds})}{\rho^2} + \frac{n(\rho-n)((\tilde{dx})'(\overline{ds}) + (\tilde{ds})'(\overline{dx}))}{\rho^2}. \tag{19}$$

If g has no root as least as large as ρ_*, then from Corollary 4.1, $g(\rho_*) < 0$ and by continuity, $g(\rho) < 0$ for all $\rho \geq \rho_*$. From Theorem 4.1, a step of length one would maintain feasibility for any choice of $\rho \geq \rho_*$. Notice that from (13) and (19), with a step of length one and for any choice for $\rho > \rho_*$,

$$x'_{k+1}s_{k+1} = \frac{n}{p}x'_k s_k + \frac{n^2(\overline{dx})'(\overline{ds})}{\rho^2} + \frac{n(\rho-n)((\tilde{dx})'(\overline{ds}) + (\tilde{ds})'(\overline{dx}))}{\rho^2};$$

hence we can get arbitrarily close to optimality by taking a unit step with a large enough value for ρ. The limiting directions as $\rho \to \infty$ are given by (7) or (9), are feasible, and are the directions used with the maximal feasible step during Step 1 of the algorithm, so the algorithm will terminate at optimality this iteration. □

THEOREM 4.3. *Suppose the* LP *or* LCP *has integer (or rational) data with encoding length L, and suppose (without loss of generality) that* (x_0, s_0) *is an approximately centered strictly feasible pair with parameter* $\beta \leq 0.5$ *such that* $\log(x'_0 s_0) = O(\sqrt{n}L)$. *Then Algorithm* I *(for* LP*) and Algorithm* II *(for* LCP*) require at most* $O(\sqrt{n}L)$ *iterations to compute feasible vectors x and s such that* $x's \leq 2^{-2L}$.

Proof. From Theorem 4.2 we can assume without loss of generality that g has a root at least as large as ρ_*, say ρ_k, during each iteration k; hence a step length of one is used each iteration. From (13), after iteration k,

$$x'_{k+1}s_{k+1} = \frac{n}{p}x'_k s_k + (dx)'_k(ds_k).$$

Now,

$$(dx)'_k(ds)_k = (dx)'_k D_k^{-1} D_k(ds)_k \leq \|D_k^{-1}(dx)_k\|_2 \|D_k(ds)_k\|_2$$
$$\leq \frac{1}{2}(\|D_k^{-1}(dx)_k\|_2^2 + \|D_k(ds)_k\|_2^2) \leq \frac{\|v_k\|^2}{2},$$

where the last inequality follows from (16).

Consequently,

$$x'_{k+1}s_{k+1} \leq \frac{n}{\rho_k} x'_k s_k + \frac{\|v_k\|^2}{2};$$

so from (15),

$$x'_{k+1}s_{k+1} \leq \frac{2n\rho_k + (\rho_k - n)^2 + n\beta^2/(1-\beta)}{2\rho_k^2} x'_k s_k$$
$$= \frac{\rho_k^2 + n^2 + n\beta^2/(1-\beta)}{2\rho_k^2} x'_k s_k$$
$$\leq \left(\frac{1}{2} + \frac{n(n + \beta^2/(1-\beta))}{2\rho_*^2}\right) x'_k s_k \leq \left(\frac{1}{2} + \frac{n(n + \beta^2/(1-\beta))}{2(n + \sqrt{\alpha}\sqrt{n})^2}\right) x'_k s_k.$$

Now, since $\beta \leq 0.5$ implies that $\beta^2/(1-\beta) \leq \alpha \leq 0.5$, we have $n + \beta^2/(1-\beta) \leq n + \sqrt{\alpha}\sqrt{n}$, where α is defined in Corollary 4.1, so

$$x'_{k+1}s_{k+1} \leq \left(\frac{1}{2} + \frac{n}{2(n + \sqrt{\alpha}\sqrt{n})}\right) x'_k s_k,$$

and since $n/(n + \sqrt{\alpha}\sqrt{n}) = 1 - \sqrt{\alpha}/(\sqrt{\alpha} + \sqrt{n})$, we have

$$x'_{k+1}s_{k+1} \leq \left(1 - \frac{\sqrt{\alpha}}{4\sqrt{n}}\right) x'_k s_k.$$

Given this rate of reduction each iteration, the theorem follows using a standard argument (see, for example, [32]). □

If x and s are feasible and $x's \leq 2^{-2L}$, it is well known that an optimal solution can be refined from x and s using at most $O(n^3)$ arithmetic operations. Consequently, Algorithms I and II require at most $O(\sqrt{n}L)$ iterations, where each iteration requires $O(n^3)$ arithmetic operations.

4.2. Superlinear convergence.

THEOREM 4.4. *If the iterates (x_k, s_k) of Algorithm* I *or Algorithm* II *converge to a strictly complementary optimal solution (x_*, s_*), then the duality gap converges to zero q-superlinearly.*

Proof. From Theorem 4.3, we already know that the duality gap converges to zero. Due to a result of Güler and Ye [6], in the LP case, we need only assume that the iterates converge to an optimal solution (x_*, s_*) because approximate centering will guarantee that the solution is strictly complementary. Consider component i of iterates x_k and s_k. Assume without loss of generality (the argument will be symmetric in x and s) that $x_k^i \to x_*^i > 0$ and $s_k^i \to 0$. For component i, the sequence $x_k^i \to x_*^i$, so

$$\frac{x_{k+1}^i}{x_k^i} \to 1, \qquad \text{and therefore} \quad 1 + \frac{(dx)_k^i}{x_k^i} \to 1 \text{ as } k \to \infty.$$

Since $x_k^i \to x_*^i > 0$, we have $(dx)_k^i \to 0$ as $k \to \infty$.

We now show that eventually the sequence $\{(ds)_k^i/\Delta_k\}$ is uniformly bounded. Since $x_k^i \to x_*^i > 0$, there exists an index K such that $k \geq K$ implies that

$$\tfrac{1}{2}x_*^i \leq x_k^i \leq 2x_*^i. \tag{20}$$

For $k \geq K$, the approximately centered condition and (20) imply that

$$\frac{(1-\beta)}{2x_*^i} \leq \frac{s_k^i}{\Delta_k} \leq \frac{2(1+\beta)}{x_*^i}. \tag{21}$$

From (16),

$$\|D_k^{-1}(dx)_k\|^2 + \|D_k(ds)_k\|^2 \leq \|v_k\|^2,$$

and from (15), $\|v_k\|^2 \leq 2n\Delta_k$; hence $\|D_k(ds)_k\| \leq \sqrt{2n\,\Delta_k}$. Therefore, for each iteration $k \geq K$,

$$|d_k^i(ds)_k^i| \leq \sqrt{2n\,\Delta_k}, \quad \text{that is,} \quad \left|\frac{(ds)_k^i}{\Delta_k}\right| \leq \sqrt{\frac{2n\,s_k^i}{x_k^i\,\Delta_k}}.$$

If we apply (20) and (21) we get

$$\left|\frac{(ds)_k^i}{\Delta_k}\right| \leq \frac{\sqrt{8(1+\beta)n}}{x_*^i},$$

which means that the sequence $\{(ds)_k^i/\Delta_k\}$ is eventually uniformly bounded.

Since $(dx)_k^i \to 0$ and $\{(ds)_k^i/\Delta_k\}$ is eventually uniformly bounded and since a similar argument holds for each i, we have

$$\frac{\|(D_x)_k(D_s)_k e\|}{\Delta_k} \to 0.$$

If the algorithm does not terminate finitely, $g(\rho_k) = 0$ each iteration. This implies $\|(D_x)_k(D_s)_k e\| > 0$ since the second term in the definition of g in (12) is strictly positive. Therefore, each iteration satisfies

$$\frac{n\beta\Delta_k}{\rho_k\|(D_x)_k(D_s)_k e\|} = 1,$$

and since $\|(D_x)_k(D_s)_k e\|/\Delta_k \to 0$, it must be that $\rho_k \to \infty$. Note also that $(dx)_k'(ds)_k/\Delta_k \to 0$. If we define t_k from

$$(dx)_k'(ds)_k := t_k(x_k's_k), \tag{22}$$

this means $t_k \to 0$. From (13),

$$x_{k+1}'s_{k+1} = \left(\frac{n}{\rho_k} + t_k\right)x_k's_k. \tag{23}$$

Since $\rho_k \to \infty$ and $t_k \to 0$, superlinear convergence follows. □

COROLLARY 4.2. *If the iterates (x_k, s_k) of Algorithm* I *or Algorithm* II *converge to a strictly complementary optimal solution (x_*, s_*) to* LP *or* LCP, *then each component of the sequence $\{X_kS_ke\}$ converges to zero q-superlinearly.*

Proof. The approximately centered condition (1) implies that each component i of the kth iterate satisfies

$$(1-\beta)\Delta_k \le x_k^i s_k^i \le (1+\beta)\Delta_k$$

and each component of the next iterate satisfies

$$(1-\beta)\Delta_{k+1} \le x_{k+1}^i s_{k+1}^i \le (1+\beta)\Delta_{k+1};$$

therefore

$$\frac{(1-\beta)\Delta_{k+1}}{(1+\beta)\Delta_k} \le \frac{x_{k+1}^i s_{k+1}^i}{x_k^i s_k^i} \le \frac{(1+\beta)\Delta_{k+1}}{(1-\beta)\Delta_k}.$$

The quantity Δ_{k+1}/Δ_k was shown to tend to zero in Theorem 4.4; hence $x_k^i s_k^i$ converges to zero q-superlinearly as k increases. □

4.3. Quadratic convergence in the nondegenerate LP case. Q-quadratic convergence in the nondegenerate LP case can be shown using Lemma 3.2 in [34] or in a manner similar to that used by Ye, Güler, Tapia, and Zhang [31], which we now describe. Assume that the LP is nondegenerate and hence has a unique, strictly complementary solution (x_*, s_*). Nondegeneracy along with the convergence of the duality gap is enough to show that the iterates converge to (x_*, s_*), so in this case we do not need to assume that the iterates converge. The matrix A can be partitioned (after suitably permuting columns) into the form

$$A = [B \quad N], \tag{24}$$

where the indices corresponding to B correspond to entries in x_* that are strictly positive, and the indices related to N correspond to entries of s_* that are strictly positive. We can similarly partition x, s, dx, and ds into x_B, s_B, x_N, s_N, dx_B, ds_B, dx_N, and ds_N. The matrix B is square and invertible if the LP is nondegenerate. For simplicity, we suppress the iteration index.

Let $\alpha > 0$ be a constant lower bound for the entries of x_B and s_N (due to approximate centering; see [6]). The "O" notation below means that the quantity under consideration is bounded by the quantity in parentheses multiplied by a constant independent of the iteration number.

LEMMA 4.1. *Suppose the* LP *is partitioned as in* (24). *Then*

$$\|dx_N\| = O(x's) \quad \text{and} \quad \|ds_B\| = O(x's).$$

Proof. Notice that $\|dx_N\| \le \|D_N\| \, \|D_N^{-1} dx_N\|$. From (15), $\|v\|^2 \le 2n\Delta$, and since from (16) $\|D^{-1}dx\|^2 + \|D\,ds\|^2 \le \|v\|^2$,

$$\|dx_N\| \le \frac{1}{\sqrt{\alpha}} \|X_N^{1/2}\| \sqrt{2n\Delta} = O(x's).$$

One can show that $\|ds_B\| = O(x's)$ similarly. □

THEOREM 4.5. *Suppose the* LP *is partitioned as in* (24) *and is nondegenerate (hence B is invertible). Then the duality gap converges to zero q-quadratically, and each component of the sequence $\{X_k S_k e\}$ converges to zero q-quadratically.*

Proof. Since $A\,dx = 0$, $\|dx_B\| = \|-B^{-1}N\,dx_N\| = O(x's)$ using Lemma 4.1. Also, since ds is a dual feasible direction, there exists a vector dy such that $A'dy + ds = 0$; that is, $B'dy + ds_B = 0$ and $N'dy + ds_N = 0$, which with Lemma 4.1 implies

$$\|ds_N\| = \|N'B'^{-1}\,ds_B\| = O(x's).$$

Together with Lemma 4.1 we have

$$\frac{\|D_x D_s e\|}{\Delta} = O(x's).$$

Assuming the algorithm does not terminate finitely, ρ_k will always be a root of g. Using the proof of Theorem 4.4 and the definition of g in (12), we see that

$$x'_{k+1}s_{k+1} \leq \frac{n}{\rho_k} x'_k s_k = \frac{\|(D_x)_k (D_s)_k e\|}{\beta \Delta_k} x'_k s_k = O((x'_k s_k)^2),$$

which establishes q-quadratic convergence. In addition, the argument of Corollary 4.2 shows that each component of the sequence $\{X_k S_k e\}$ converges to zero q-quadratically. □

4.4. Quadratic convergence in the nondegenerate LCP case. If we assume that the LCP has a unique, strictly complementary solution (x_*, s_*), quadratic convergence can be easily demonstrated using some of the ideas found in Ye and Anstreicher [30]. Using their notation, the matrix M can be partitioned (by suitably rearranging rows and columns) into the form

$$M = \begin{bmatrix} M_{BB} & M_{BN} \\ M_{NB} & M_{NN} \end{bmatrix}, \tag{25}$$

where the indices B refer to the entries in x_* that are strictly positive and the indices N refer to the entries in s_* that are strictly positive. We will suppress the iteration index for simplicity.

LEMMA 4.2. *Suppose M is partitioned as in* (25). *Then* $\|dx_N\| = O(x's)$ *and* $\|ds_B\| = O(x's)$.

Proof. From (4), $S_B^{-1}\, ds_B = (X_B S_B)^{-1/2} v_B - X_B^{-1}\, dx_B = n\Delta/\rho (X_B S_B)^{-1} e - e - X_B^{-1}\, dx_B$. The approximately centered condition implies that $\Delta/(x^i s^i)$ is bounded above by a constant, and Theorem 4.4 shows that $\rho \to \infty$ and $\|dx_B\| \to 0$. Since each component of x_B is eventually bounded away from zero by a constant value, $S_B^{-1}\, ds_B \to -e$. Consequently, each component of ds_B will eventually become strictly negative with $-ds_B < s_B$ (since a unit step will be taken), which implies $\|ds_B\| \leq \|s_B\| = O(x's)$. One can show that $\|dx_N\| = O(x's)$ similarly. □

THEOREM 4.6. *If there is a unique, strictly complementary solution (x_*, s_*), to* LCP, *then the iterates (x_k, s_k) of Algorithm* II *converge to (x_*, s_*), and the duality gap converges to zero q-quadratically. Also, each component of the sequence $\{X_k S_k e\}$ converges to zero q-quadratically.*

Proof. The convergence of the iterates follows from the fact that the duality gap converges to zero. In order to show q-quadratic convergence of the duality gap to zero, note that uniqueness of the solution implies that M_{BB} is invertible. If we use (4) and (25),

$$\|dx_B\| = \|M_{BB}^{-1}(ds_B - M_{BN} dx_N)\| = O(x's)$$

and

$$\|ds_N\| = \|M_{NB} dx_B + M_{NN} dx_N\| = O(x's),$$

so with Lemma 4.2, $\|dx\| = O(x's)$ and $\|ds\| = O(x's)$; hence, $t_k = O(x's)$ (where t_k is defined in (22)) and $\|D_x D_s e\|/\Delta = O(x's)$, which implies that $n/\rho_k = O(x's)$. Quadratic convergence follows from (23). The componentwise quadratic convergence of $\{X_k S_k e\}$ to zero can be shown as in Corollary 4.2. □

5. Concluding remarks. Many interesting questions concerning Algorithms I and II, and other similar algorithms, remain. Can these or any variants which use one factorization per iteration be shown to converge q-quadratically in the degenerate case? How well do the algorithms perform in practice? (See [20] for promising computational results with algorithms quite similar to Algorithms I and II.) Can similar algorithms incorporate a combined Phase I-Phase II scheme or work without the approximately centered condition as potential reduction algorithms do? Do the primal and dual iterates converge to optimal solutions if degeneracy exists?

Improvements to Algorithms I and II are possible. The use of larger values for ρ, possibly calculated by a line search rather than as the root of a polynomial, which allow unit steps and maintain approximate centering would result in an algorithm with the same local and global properties as Algorithms I and II and probably would perform better in practice. An LCP algorithm based on Todd's projected, scaled, and steepest descent directions [23] would probably be theoretically superior to Algorithm II. The use of larger values for β is also possible.

Acknowledgments. The author thanks Mike Todd of Cornell University for his useful comments on the various versions of this paper and three anonymous referees for significantly improving the paper's presentation.

REFERENCES

[1] D. A. Bayer and J. C. Lagarias, *The nonlinear geometry of linear programming I. Affine and projective scaling trajectories*, Trans. Amer. Math. Soc., 314 (1987), pp. 499–526.

[2] ———, *The nonlinear geometry of linear programming II. Legendre transform coordinates and central trajectories*, Trans. Amer. Math. Soc., 314 (1987), pp. 527–581.

[3] T. F. Coleman and Y. Li, *A quadratically-convergent algorithm for the linear programming problem with lower and upper bounds*, in Large-Scale Numerical Optimization, T. Coleman and Y. Li, eds., Society for Industrial and Applied Mathematics, Philadelphia, PA, 1990, pp. 49–57.

[4] R. M. Freund, *Polynomial-time algorithms for linear programming based only on primal scaling and projected gradients of a potential function*, Math. Programming, 52 (1991), pp. 203–222.

[5] C. C. Gonzaga, *An algorithm for solving linear programming problems $O(n^3L)$ operations*, in Progress in Mathematical Programming: Interior-Point and Related Methods, N. Megiddo, ed., Springer-Verlag, Berlin, 1989, pp. 1–28.

[6] O. Güler and Y. Ye, *Convergence behavior of some interior-point algorithms*, Working paper 91-4, The College of Business Administration, The University of Iowa, Iowa City, IA, 1991.

[7] M. Iri and H. Imai, *A multiplicative barrier function method for linear programming*, Algorithmica, 1 (1986), pp. 455–482.

[8] J. Ji, F. Potra, and S. Huang, *A predictor-corrector method for linear complementarity problems with polynomial complexity and superlinear convergence*, Working Paper 91-18, Department of Mathematics, The University of Iowa, Iowa City, IA, 1991.

[9] J. Ji, F. Potra, R. A. Tapia, and Y. Zhang, *An interior-point algorithm with polynomial complexity and superlinear convergence*, Department of Mathematics, The University of Iowa, Iowa City, IA, 1991, manuscript.

[10] M. Kojima, N. Megiddo, and S. Noma, *Homotopy continuation methods for nonlinear complementarity problems*, Math. Oper. Res., 16 (1991), pp. 754–774.

[11] M. Kojima, S. Mizuno, and A. Yoshise, *A polynomial-time algorithm for a class of linear complementarity problems*, Math. Programming, 44 (1989), pp. 1–26.

[12] ———, *An $O(\sqrt{n}L)$ iteration potential reduction algorithm for linear complementarity problems*, Math. Programming, 50 (1991), pp. 331–342.

[13] N. Megiddo, *Pathways to the optimal set in linear programming*, in Progress in Mathematical Programming: Interior Point and Related Methods, N. Megiddo, ed., Springer-Verlag, New York, 1989, pp. 131–158.

[14] N. Megiddo and M. Shub, *Boundary behaviour of interior point algorithms for linear programming*, Math. Oper. Res., 14 (1989), pp. 97–146.

[15] S. Mehrotra, *Finite termination and superlinear convergence in primal-dual methods*, Tech. report 91-13, Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL, 1991.

[16] S. MEHROTRA, *Quadratic convergence in a primal-dual method*, Tech. report 91-15, Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL, 1991.

[17] J. E. MITCHELL AND M. J. TODD, *On the relationship between the search directions in the affine and projective scaling variants of Karmarkar's algorithm*, in Contributions to O.R. and Economics, B. Cornet and H. Tulkens, eds., MIT Press, Cambridge, MA, 1989, pp. 237–250.

[18] S. MIZUNO AND M. J. TODD, *An $O(n^3L)$ adaptive path following algorithm for a linear complementarity problem*, Math. Programming, 52 (1991), pp. 587–595.

[19] S. MIZUNO, M. J. TODD, AND Y. YE, *On adaptive-step primal-dual interior-point algorithms for linear programming*, Math. Oper. Res. 18 (1993), pp. 964–981.

[20] S. MIZUNO, A. YOSHISE, AND T. KIKUCHI, *Practical polynomial time algorithms for linear complementarity problems*, J. Oper. Res. Soc. Japan, 32 (1989), pp. 75–92.

[21] R. D. C. MONTEIRO AND I. ADLER, *Interior path-following primal-dual algorithms. Part I: linear programming*, Math. Programming, 44 (1989), pp. 27–41.

[22] J. RENEGAR, *A polynomial time algorithm, based on Newton's method, for linear programming*, Math. Programming, 40 (1988), pp. 59–94.

[23] M. J. TODD, *Projected scaled steepest descent in Kojima-Mizuno-Yoshise's algorithm for the linear complementarity problem*, Tech. report 950, School of Operations Research and Industrial Engineering, Cornell University, Ithaca, NY, 1989.

[24] M. J. TODD AND Y. YE, *A centered projective algorithm for linear programming*, Math. Oper. Res., 15 (1990), pp. 508–529.

[25] T. TSUCHIYA, *Quadratic convergence of Iri and Imai's algorithm for degenerate linear programming problems*, manuscript, The Institute of Statistical Math., Tokyo, Japan, 1991.

[26] T. TSUCHIYA AND K. TANABE, *Local convergence properties of new methods in linear programming*, J. Oper. Res. Soc. Japan, 33 (1990), pp. 22–45.

[27] H. YAMASHITA, *A polynomially and quadratically convergent method for linear programming*, manuscript, Mathematical Systems Institute, Tokyo, Japan, 1986.

[28] Y. YE, *Further development on the interior algorithm for convex quadratic programming*, Tech. report, Integrated Systems Inc., Santa Clara, CA, and Dept. of Engineering-Economic Systems, Stanford University, Stanford, CA, 1987.

[29] ———, *An $O(n^3L)$ potential reduction algorithm for linear programming*, Math. Programming, 50 (1991), pp. 239–258.

[30] Y. YE AND K. M. ANSTREICHER, *On quadratic and $O(\sqrt{n}L)$ convergence of a predictor-corrector algorithm for LCP*, manuscript, Department of Management Sciences, The University of Iowa, Iowa City, IA, 1991.

[31] Y. YE, O. GÜLER, R. A. TAPIA, AND Y. ZHANG, *A quadratically convergent $O(\sqrt{n}L)$-iteration algorithm for linear programming*, Working paper Series 91-14, The University of Iowa, Iowa City, IA, 1991.

[32] Y. YE, R. A. TAPIA, AND Y. ZHANG, *A superlinearly convergent $O(\sqrt{n}L)$-iteration algorithm for linear programming*, Tech. report TR91-22, Department of Mathematical Sciences, Rice University, Houston, TX, 1991.

[33] Y. ZHANG AND R. A. TAPIA, *A superlinearly convergent polynomial primal-dual interior-point algorithm for linear programming*, SIAM J. Optimization, 3 (1993), pp. 118–133.

[34] Y. ZHANG, R. A. TAPIA, AND J. E. DENNIS, *On the superlinear and quadratic convergence of primal-dual interior-point linear programming algorithms*, SIAM J. Optim., 2 (1992), pp. 304–324.

[35] Y. ZHANG, R. A. TAPIA, AND F. POTRA, *On the superlinear convergence of interior-point algorithms for a general class of problems*, Tech. report TR90-9, Department of Mathematical Sciences, Rice University, Houston, TX, 1990.

SIAM J. OPTIMIZATION
Vol. 4, No. 2, pp. 262–287, May 1994
© 1994 Society for Industrial and Applied Mathematics
003

CONVERGENCE PROPERTIES OF A CLASS OF RANK-TWO UPDATES*

PAUL T. BOGGS† AND JON W. TOLLE‡

Abstract. Many optimization algorithms generate, at each iteration, a pair (x_k, H_k) consisting of an approximation to the solution x_k and a Hessian matrix approximation H_k that contains local second-order information about the problem. Much is known about the convergence of x_k to the solution of the problem, but relatively little is known about the behavior of the sequence of matrix approximations. The sequence $\{H_k\}$, generated by the extended Broyden class of updating schemes independently of the optimization setting in which they are used, is analyzed. Various conditions under which convergence is assured are derived, and the structure of the limits is delineated. Rates of convergence are also obtained. These results extend and clarify those already in the literature.

Key words. optimization, quasi-Newton algorithms, matrix updates, convergence analysis

AMS subject classifications. 90C30, 65K05

1. Introduction.

We investigate a general class of matrix updating schemes that are of major interest in optimization problems, both constrained and unconstrained. We consider the possible convergence of sequences generated by these schemes, attempting to isolate the crucial factors that determine convergence independent of any particular optimization setting. The results presented thus subsume and clarify some recent results on convergence of Hessian approximations in specific optimization algorithms.

In many iterative numerical optimization algorithms each iteration requires the solution (or partial solution) of an approximate quadratic model of the problem under consideration. Each surrogate model is usually characterized by the current iterate x_k and a symmetric, often positive definite, matrix H_k. The solution of the quadratic model yields a step s_k from which the next iterate, $x_{k+1} = x_k + s_k$, is obtained. Once the new iterate is located a new quadratic model, i.e., a new matrix, H_{k+1}, is chosen. The choice is determined by a particular formula involving the information on hand, including the preceding matrix H_k, the step s_k, and the values of the problem functions at x_k and x_{k+1}. The success of the algorithm is known to be sensitive in many cases to the method by which the new matrix is chosen; in particular, the best results are to be expected when the quadratic model closely reflects the important features of the underlying problem as the solution is approached. For this reason the convergence properties of the sequence $\{H_k\}$ are of significant interest. This paper is devoted to the study of these properties.

We examine an iteration scheme of the form

$$H_{k+1} = \mathcal{U}(\phi_k, H_k, s_k, y_k), \tag{1.1}$$

where the H_k are $N \times N$ symmetric matrices, the ϕ_k are scalar parameters, the s_k are given vectors in $\mathcal{R}^N$, the y_k are vectors that depend on the s_k in some way to be specified, and the function $\mathcal{U}$ is the update formula. Of particular interest will be how the convergence (and rate of convergence) of the sequence $\{H_k\}$ depends on the choice of the ϕ_k, the initial matrix H_0, and the sequence $\{s_k\}$ for a given $\mathcal{U}$.

The primary motivation for this study is the use of these updates in those optimization algorithms where the function $\mathcal{U}$ adds a rank-two matrix to the current matrix H_k in such a

* Received by the editors March 13, 1991; accepted for publication (in revised form) March 24, 1993. (JONTOLLE@UNC.EDU).

† National Institute for Standards and Technology, Gaithersburg, Maryland 20899. This author's research was supported by Air Force Office of Scientific Research grant AFOSR-ISSA-90-0004.

‡ Departments of Mathematics and Operations Research, University of North Carolina, Chapel Hill, North Carolina 27599. This author's research was supported by Air Force Office of Scientific Research grant AFOSR-88-0267.

way that the secant condition

$$H_{k+1}s_k = y_k \tag{1.2}$$

is satisfied for each k. The best known example of the use of this updating method takes place in quasi-Newton algorithms for solving unconstrained optimization problems where the matrix H_k is an approximation to the Hessian of the objective function. In this case s_k is the step generated by

$$s_k = -\alpha_k H_k^{-1} \nabla f(x_k) \tag{1.3}$$

and

$$y_k = \nabla f(x_k + s_k) - \nabla f(x_k). \tag{1.4}$$

Here f is the objective function, x_k is the current iterate, and α_k is a steplength parameter. Similar updating schemes are also used in trust region methods for the unconstrained problem and for the sequential quadratic programming algorithms designed to solve constrained optimization problems.

The most commonly used class of updating functions in these cases is the so-called "extended Broyden" class where

$$\begin{aligned} \mathcal{U}(\phi_k, H_k, s_k, y_k) = &\left(I - \frac{y_k s_k^t}{y_k^t s_k}\right) H_k \left(I - \frac{s_k y_k^t}{y_k^t s_k}\right) + \frac{y_k y_k^t}{y_k^t s_k} \\ &- \phi_k(s_k^t H_k s_k) \left(\frac{y_k}{y_k^t s_k} - \frac{H_k s_k}{s_k^t H_k s_k}\right) \left(\frac{y_k}{y_k^t s_k} - \frac{H_k s_k}{s_k^t H_k s_k}\right)^t. \end{aligned} \tag{1.5}$$

The special cases $\phi_k = 0$ and $\phi_k = 1$ correspond to the well-known DFP and BFGS update formulas, respectively, while the values of $\phi_k \in [0, 1]$ give rise to the "restricted Broyden class" of updates. There is also a similar set of updates in which s_k and y_k are interchanged; these updating schemes yield approximations to the inverse Hessian. (See Dennis and Schnabel [1] or Fletcher [2].)

There has been a significant amount of research on the convergence of the matrices generated by these updating schemes, most generally for the case described above in which each s_k is a step in an optimization algorithm. For example, Schuller [3], Ge and Powell [4], and Stoer [5] have determined conditions under which the sequence of matrices converge in the setting of variable metric algorithms where each s_k is given by (1.3). More recently, Byrd, Nocedal, and Yuan [6] have studied the Broyden class of updates seeking to explain the observed superiority of the BFGS update over the DFP update in quasi-Newton algorithms. Conn, Gould, and Toint [7] have analyzed the special case of (1.5) where $\mathcal{U}(\phi_k, H_k, s_k, y_k)$ is the symmetric rank-one update. More significantly, they do not require that s_k be generated by (1.3) but only that it be the step $x_{k+1} - x_k$. (However, they do take the vector y_k to be the difference in the gradients at x_{k+1} and x_k.) As they point out, this more general formulation allows the analysis to include the important trust region algorithms as well as large-scale methods that use partitioned matrices.

The results presented herein are in the spirit of the approach in [7] in that the s_k are not required to satisfy (1.3); however, we go a step further and disassociate them from any particular optimization problem, although the setting is motivated by unconstrained optimization. We will establish the convergence of $\{H_k\}$ under only minimal restrictions on $\{s_k\}$ that are not related directly to its generation by an optimization process. Moreover, we will not require the y_k to satisfy (1.4) but rather that they be related to the s_k in a certain way that includes (1.4) as

a special case. We also will show how the structure of the limit of the sequence $\{H_k\}$ depends upon the properties of the sequence $\{s_k\}$, thus clarifying the results in [3]–[7]. Finally, our results can be applied to most of the common rank-two updating schemes that have been used in unconstrained optimization.

In §2 the nuances of the problem to be solved are discussed and a simplified framework in which to analyze the convergence is introduced. Specifically, we consider the class of updates as nonlinear perturbations of a single linear update that derives from setting $\phi_k = 0$ in (1.5). Section 3 contains some basic notation and terminology as well as some fundamental results on difference equations that are used in the remainder of the paper. In §4 the convergence of the linear update is established for the case when the sequence $\{s_k\}$ repeatedly spans $\mathcal{R}^N$ in a uniform manner and $y_k = s_k$ for each k. As noted in §2, this latter condition can be assumed without loss of generality in the unconstrained optimization of a convex quadratic function. In §5 this result is generalized to allow the s_k to approach a proper subspace of $\mathcal{R}^N$. In particular, the counterexample in [4] to general convergence of the sequence $\{H_k\}$ is clarified. In §6 the convergence results are extended to nonlinear updating schemes, i.e., choices of ϕ_k other than zero.

Finally, in §7 the condition $y_k = s_k$ is relaxed, permitting the previously obtained results to be applied to the updates of the form of (1.5). The basic requirement is only that the difference $y_k - s_k$ converge to zero at a specified rate. Section 8 contains a summary of the major conclusions in the paper, observations on their relevance, and suggestions for further avenues of research.

2. Problem formulation. In the quasi-Newton algorithm for unconstrained optimization problems $s_k = x_{k+1} - x_k$ and y_k is given by (1.4). Assuming that f is sufficiently smooth, we can write

$$y_k = F(x_k)s_k + O(|s_k|^2) = F(x^*)s_k + (F(x_k) - F(x^*))s_k + O(|s_k|^2),$$

where $F(x)$ is the Hessian of f at x and x^* is a minimum point of f. Thus

$$y_k = F(x^*)s_k + \xi_k,$$

where $\xi_k/|s_k| \to 0$ provided $x_k \to x^*$. If $F = F(x^*)$ is positive definite, then, since the form of the update (1.5) is invariant under the transformations,

$$\text{(T)} \qquad s \leftrightarrow F^{-1/2}s, \quad y \leftrightarrow F^{1/2}y, \quad \text{and} \quad H \leftrightarrow F^{1/2}HF^{1/2},$$

$F(x^*)$ can, without loss of generality, be assumed to be the identity matrix. Using this case as motivation, the underlying assumption for our analysis of (1.5) will be that

$$y = s_k + \xi_k, \tag{2.1}$$

where $\xi_k/|s_k| \to 0$ at a rate to be specified. Note that if the objective function is quadratic then $\xi_k = 0$. Initially, we will analyze this special case. Then we will allow the more general formula (2.1) as a perturbation of this simpler situation. Further comments on this assumption are made in the course of the paper.

With the assumption that $y_k = s_k$, the update scheme generated by (1.5) no longer depends on the length of s_k and can be written

$$\begin{aligned} \hat{H} &= \mathcal{U}(\phi, H, s, s) \\ &= (I - ss^t)H(I - ss^t) + ss^t - \phi(s^tHs)\left[s - \frac{Hs}{s^tHs}\right]\left[s - \frac{Hs}{s^tHs}\right]^t, \end{aligned} \tag{2.2}$$

where the subscript k has been deleted, $\hat{H}$ has replaced H_{k+1}, and $|s| = 1$. It is observed that both the Broyden class of update formulas for the approximations to the inverse Hessian and to the Hessian satisfy this equation when $y = s$. In fact, (2.2) represents the most general rank-two symmetric update formula that uses the vectors s and Hs and satisfies the secant equation (1.2) (which is now $\hat{H}s = s$). Thus (2.2) represents a rather general starting point for analyzing the convergence properties of rank-two updates. It should be noted that other rank-two update formulas, such as the well-known PSB update, that are not members of the Broyden class may not be able to be placed in this form because they are not invariant under the transformations (T); that is, it cannot be assumed that the Hessian of the objective function is the identity matrix. However, as is noted in §7, in certain cases the convergence of the updates may still be analyzed by the methods we develop.

The two special cases identified in §1 are singled out for emphasis: $\phi = 0$, which yields

$$\text{(L)} \qquad \hat{H} = (I - ss^t)\, H(I - ss^t) + ss^t,$$

and $\phi = 1$, which gives

$$\text{(C)} \qquad \hat{H} = H + ss^t - \frac{Hss^tH}{s^tHs}.$$

These two updating formulas, which we will hereafter refer to as the "L update" (L for linear) and the "C update" (for complementary), can be thought of as versions of the well-known DFP and BFGS updating formulas. The reader should be careful to note that the BFGS, DFP, and other updates are not uniquely identified in the formula (2.2) because the assumption that $y = s$ removes the reference to the particular optimization setting that might generate s. For example, the L update is derived from the DFP update for the approximation of the Hessian matrix or, alternatively, the BFGS update for approximating the inverse Hessian matrix. Similarly, the C update derives from the direct BFGS or inverse DFP update.

We will first be concerned with the convergence of the sequence $\{H_k\}$ generated by the iterative scheme (2.2) for arbitrary positive definite starting matrices H_0 (although many of our results will apply for arbitrary symmetric matrices) and specified sequences of unit vectors $\{s_k\}$. Two types of methods of choosing the parameter ϕ_k can be considered: a "static" procedure where ϕ_k is constant over all iterations and a "dynamic" method where the choice of ϕ_k will vary from iteration to iteration and may depend on the current values of s_k and H_k. (If the dynamic method is derived from the original update formula (1.5) then it must be invariant under the transformations given above if it is to be considered here.) The static method is typified by the L ($\phi_k = 0$ for all k) and the C ($\phi_k = 1$ for all k) updates and the dynamic method by the "symmetric rank-one" update discussed below. In theory, the dynamic approach can be used to take better advantage of the current information and can therefore lead to more rapid convergence of the sequence $\{H_k\}$. This is illustrated by the aforementioned symmetric rank-one scheme in which the choice of ϕ_k leads to convergence of the $\{H_k\}$ in a finite number ($\leq N$) of steps whenever the vectors $s_0, \ldots, s_{N-1}$ are linearly independent.

It is useful to consider the inverse corresponding to the update (2.2). Assuming that H and $\hat{H}$ are invertible, an application of the Sherman–Morrison–Woodbury formula yields

$$\hat{H}^{-1} = \mathcal{U}(\mu(\phi),\, H^{-1}, s, s), \tag{2.3}$$

where

$$\mu(\phi) = \frac{(1-\phi)(s^tHs)(s^tH^{-1}s)}{\phi + (1-\phi)(s^tHs)(s^tH^{-1}s)}.$$

It is observed that $\mu(0) = 1$ and $\mu(1) = 0$, illustrating the property that the inverse of the L update is the C update of the inverse and vice versa. Moreover, if H is positive definite, then $(s^tHs)(s^tH^{-1}s) \geq 1$ (with equality if and only if s is an eigenvector of H) and it is seen that $0 < \phi < 1$ implies that $\mu(\phi)$ falls between 0 and 1. Thus the inverse of a restricted Broyden update of a positive definite matrix can be obtained by a (dynamic) restricted Broyden update of the inverse of the matrix.

It should also be observed that if H is positive definite, then so is H^{-1} and thus, since the L and C updates preserve positive definiteness, (2.2) and (2.3) imply, by the interleaving eigenvalues theorem (see [8]), that $\hat{H}$ and $\hat{H}^{-1}$ are positive definite if either ϕ or $\mu(\phi)$ is less than or equal to zero. More specifically we have Proposition 2.1.

PROPOSITION 2.1 (Fletcher [2]). *Let H be positive definite and assume that s is not an eigenvector of H. Then the iterate $\hat{H}$ given by* (2.2) *is positive definite if and only if $\phi < \phi_C$, where*

$$\phi_C = -\frac{(s^tHs)(s^tH^{-1}s)}{1 - (s^tHs)(s^tH^{-1}s)}.$$

Proof. Since $(s^tHs)(s^tH^{-1}s) > 1$ clearly $\phi_C > 1$. If the inequality holds, then either $\phi < 1$ or $\mu(\phi) \leq 0$ and by the remarks above $\hat{H}$ is positive definite. If $\phi = \phi_C$, then $\hat{H}$ is singular; in particular, $z = H^{-1}s - (s^tH^{-1}s)s$ is a null vector of $\hat{H}$. From (2.2) it follows that if $\hat{H}$ is positive definite for $\phi = \phi_1$ and for $\phi = \phi_2 > \phi_1$, then it is positive definite in (ϕ_1, ϕ_2). Thus $\hat{H}$ cannot be positive definite for any $\phi > \phi_C$. □

The case where $\phi = \phi_S = -s^tHs/(1 - s^tHs)$ is of special interest because the rank-two update formula (2.2) reduces to the rank-one update.

$$\text{(SR1)} \qquad \hat{H} = H + \frac{(s - Hs)(s - Hs)^t}{s^t(s - Hs)}$$

known as the "symmetric rank-one" (SR1) update. Note that the update formula for $\hat{H}^{-1}$ is identical to that of $\hat{H}$ for the SR1 update (with H^{-1} replacing H). If H is positive definite, the SR1 update is never a restricted Broyden update because ϕ_S is either greater than one (if $s^tHs > 1$) or less than zero (if $s^tHs < 1$). Moreover, $\hat{H}$ may not be positive definite when H is positive definite if $\phi_S > 1$; in fact, the SR1 update will not exist if $(I - H)s$ is orthogonal to s. It follows from Proposition 2.1 and the definition of ϕ_S that the SR1 update is positive definite if and only if either $s^tHs < 1$ ($\phi_S < 0$) or $s^tH^{-1}s < 1$ ($1 < \phi_S < \phi_C$).

The convergence of the static schemes corresponding to the L and C updates are interrelated as shown in the following proposition.

PROPOSITION 2.2. *Let $\mathcal{P}$ be any subset of the symmetric matrices for which $H \in \mathcal{P}$ implies $H^{-1} \in \mathcal{P}$. If for a given sequence $\{s_k\}$, the sequence $\{H_k\}$ generated by the* L *update converges to a matrix in $\mathcal{P}$ for every initial matrix chosen from $\mathcal{P}$, then the same is true for the* C *update, and conversely.*

Proof. Suppose that the sequence generated by the L updating formula converges to $\mathrm{L}(H_0) \in \mathcal{P}$. Then the sequence $\{H_k^{-1}\}$ converges to $\mathrm{L}(H_0)^{-1}$. But the sequence $\{H_k^{-1}\}$ is generated by the C updating formula starting from H_0^{-1} and so the C formula yields convergent sequences starting in $\mathcal{P}$. □

It is well known that for any choice of linearly independent s_k, $k = 0, \ldots, N - 1$, and any initial H_0, the H_k generated by the SR1 update satisfy the "hereditary" secant condition

$$\text{(2.4)} \qquad H_{k+1}s_j = s_j, \qquad j = 0, \ldots, k.$$

It is easily shown by induction that this condition is also satisfied by all updates of the form (2.2) if the s_k are orthogonal (corresponding to conjugate directions in the quadratic optimization

setting). The condition (2.4) implies that the H_k converge in at most N steps and if the full N steps are required, then the limiting matrix is the identity. Thus the iteration matrices converge finitely for any sequence $\{\phi_k\}$ when the s_k are orthogonal and for the SR1 updates when the s_k are linearly independent. In both cases the limiting matrix is the identity matrix and so it is natural to expect that, generally, if the matrices converge nonfinitely, they will also converge to the identity matrix (other possibilities are considered in §5). Accordingly, it is somewhat simpler to consider the convergence of the sequence of matrices $\{E_k\}$ where $E_k = H_k - I$. Using the relations $s_k^t s_k = 1$ and $H_k s_k = E_k s_k + s_k$ in (2.2), the E_k can be seen to satisfy the difference equation

$$\begin{aligned} E_{k+1} = {} & (1 - s_k s_k^t) E_k (I - s_k s_k^t) \\ & - \phi_k \left[\frac{1}{1 + s_k^t E_k s_k} \right] [(s_k^t E_k s_k) s_k - E_k s_k][(s_k^t E_k s_k) s_k - E_k s_k]^t. \end{aligned} \tag{2.5}$$

It is this form of the iteration scheme that we shall analyze first. Note that the quasi-Newton condition for the E_k becomes $E_{k+1} s_k = 0$ so that the matrices E_k, $k \geq 1$, are always singular and hence never positive definite; however, they are symmetric if E_0 is symmetric. The eigenvectors of H_k and E_k are identical and the eigenvalues of H_k are exactly one unit greater than the eigenvalues of E_k. Thus if the H_k are positive definite, the eigenvalues of the E_k are greater than negative one, and conversely. As a result, (2.5) is well defined provided positive definiteness of the H_k is maintained, as in the L and C updates. However, if the matrix H_k is not positive definite, then $s_k^t E_k s_k + 1$ can be zero and the resulting E_{k+1} undefined.

It is a well known fact that both the L and C updating formulas (and therefore those of the restricted Broyden class) lead to sequences $\{E_k\}$, whose Frobenius norms are nonincreasing and, in fact, are strictly decreasing if $E_k s_k \neq 0$ (see, for instance, [4] or [5]). However, as illustrated in [4], this does not ensure convergence of the sequence $\{E_k\}$ nor does it yield the limiting matrix when the sequence does converge. It should be noted that not all updates of the form (2.5) lead to sequences $\{E_k\}$ that decrease in the Frobenius norm. For example, the Frobenius norms of the matrices E_k generated by the SR1 update can have large jumps in value—although convergence within N steps is assured. The analysis of the next sections is intended to shed light on the convergence of these matrices and their limits.

3. Notation and preliminary results. Throughout this paper the notation $|x|$ will refer to the Euclidean norm of the vector x. For a matrix A, $\|A\|$ will denote the operator norm of A, i.e.,

$$\|A\| = \sup \left\{ \frac{|Ax|}{|x|} \right\},$$

while $\|A\|_F$ will denote the Frobenius norm,

$$\|A\|_F = \left(\sum (a_{ij})^2 \right)^{1/2}.$$

In referring to rates of convergence the term "m-step linear convergence" will mean Q-order convergence; that is, the sequence $\{x_k\}$ converges to zero with an m-step linear rate if there is a β, $0 \leq \beta < 1$, independent of k such that

$$|x_{k+m} - x^*| \leq \beta |x_k - x^*|$$

for all k sufficiently large. One-step linear convergence will be called simply "linear convergence." We will also refer to R-(order) linear convergence, which requires

$$(|x_k - x^*|)^{1/k} \leq \beta < 1$$

for all k sufficiently large. The important implications for our work are that m-step linear convergence implies R-linear convergence and that the components of a vector converge R-linearly if and only if the vector itself converges R-linearly (which does not hold for m-step linear convergence). It will also be useful to note that a sequence converges R-linearly if and only if it is majorized by a sequence converging (one-step) linearly. That is, $\{x_k\}$ converges R-linearly if and only if there are constants ν and τ, $0 \leq \tau < 1$, such that for all k

$$|x_k - x^*| \leq \nu \cdot \tau^k.$$

More complete information on the definitions and properties of Q- and R-order convergence can be found in [9].

Many of the proofs in the following sections will depend upon the properties of solutions of difference equations. To simplify the presentations in §§4–7 we provide in this section some useful theorems on such systems.

Consider the system of nonlinear difference equations

$$\begin{aligned} z_{k+1} &= A_k z_k + Z_k(z_k, w_k, t_k), \\ w_{k+1} &= w_k + W_k(z_k, w_k, t_k), \end{aligned} \tag{3.1}$$

where $z_k \in \mathcal{R}^p$, $w_k \in \mathcal{R}^q$, $t_k \in \mathcal{R}^n$, and A_k is a $p \times p$ matrix. Denote, for any fixed positive integer m,

$$\beta_{k,m} = \left\| \prod_{i=0}^{m-1} (A_{k+i}) \right\|. \tag{3.2}$$

The following give convergence results for the solution to (3.1) under a variety of conditions on the matrix A_k and the functions Z_k and W_k.

THEOREM 3.1. *Suppose that* $\{z_k\}$ *and* $\{w_k\}$ *satisfy* (3.1) *and, in addition,*

(i) $\beta_{k,1} \leq 1$, *i.e.,* $\|A_k\| \leq 1$, *for every* k;

(ii) *for some fixed integer* m, *there is a constant* $\beta < 1$ *and an infinite sequence of integers* $\{k_j\}$ *such that* $\beta_{k_j,m} \leq \beta$ *for each* j;

(iii) *there exist constants* κ_1 *and* κ_2 *independent of* k *such that* $|Z_k(z,w,t)| \leq \kappa_1 \cdot |t|$ *and* $|W_k(z,w,t)| \leq \kappa_2 \cdot |t|$ *for every* k; *and*

(iv) *the sequence* $\{t_k\}$ *converges to zero* R-*linearly.*

Then for any w_0 *there exists a vector* w^* *such that* $\{w_k\} \to w^*$ R-*linearly and for any* z_0 *the sequence* $\{z_k\}$ *converges to the zero vector. If condition* (ii) *is replaced by*

(ii)′ *for some fixed integer* m, *there is a constant* $\beta < 1$ *such that* $\beta_{k,m} \leq \beta$ *for every* k *sufficiently large;*

then the convergence rate of $\{z_k\}$ *is* R-*linear and if* $Z_k = 0$ *for each* k *the rate is* m-*step linear.*

THEOREM 3.2. *Suppose that conditions* (i), (ii)′, *and* (iv) *of Theorem* 3.1 *hold and, in addition,*

(iii)′ *there are positive constants* $\kappa_1, \kappa_2, \kappa_3, \kappa_4$, *and* δ *such that for each* k

$$|Z_k(z,w,t)| \leq \kappa_1 |z|^2 + \kappa_2(|z| + |w|) \cdot |t|$$

and

$$|W_k(z,w,t)| \leq \kappa_3 |z|^2 + \kappa_4(|z| + |w|) \cdot |t|,$$

whenever $|z| < \delta$.

Then there exist positive constants η_0 and τ_0 and a $w^* \in \mathcal{R}^q$ such that if $|z| < \eta_0$ and $|t_k| < \tau_0$ for every k, then the solutions to (3.1) *have the properties that $\{z_k\} \to 0$ and $\{w_k\} \to w^*$ and the convergence rates are* R*-linear. If, in particular, $\kappa_2 = 0$, then the convergence rate of $\{z_k\}$ is m-step linear.*

Sketches of the proofs of these two theorems can be found in the Appendix.

4. Convergence of the L updating method. When $\phi_k = 0$ (which corresponds to the L update) the iterations (2.5) reduce to

$$E_{k+1} = (1 - s_k s_k^t)\, E_k (I - s_k s_k^t). \tag{4.1}$$

Note that for fixed s_k (4.1) defines a linear transformation from the space of $N \times N$ matrices into itself. The other updates of the Broyden class ($\phi_k \neq 0$) can in fact be considered to be nonlinear perturbations of this linear mapping. We will exploit this fact in §6 to derive convergence results for the cases where $\phi_k \neq 0$. Denoting the mapping defined by (4.1) as

$$G(s_k; \cdot)\colon L(\mathcal{R}^N, \mathcal{R}^N) \to L(\mathcal{R}^N, \mathcal{R}^N),$$

we have that

$$E_{k+1} = G(s_k;\, E_k). \tag{4.2}$$

Since the mapping preserves symmetry, $\mathcal{S}$, the set of symmetric $N \times N$ matrices, is a $\frac{1}{2}N(N+1)$-dimensional invariant subspace for $G(s_k; \cdot)$. If we assign the Frobenius norm to $L(\mathcal{R}^N, \mathcal{R}^N)$ so that it is an inner product space, then, as the following proposition shows, the mapping $G(s; \cdot)$ is a projection.

PROPOSITION 4.1. *Let s be a given unit vector. The mapping $G(s; \cdot)$ is a projection onto an $(N-1)^2$-dimensional subspace of $L(\mathcal{R}^N, \mathcal{R}^N)$. An orthonormal set of eigenvectors corresponding to the eigenvalue zero is*

$$ss^t, s(v^2)^t, s(v^3)^t, \ldots, s(v^N)^t, v^2 s^t, \ldots, v^N s^t,$$

and an orthonormal set of eigenvectors corresponding to the eigenvalue one is

$$v^2(v^2)^t, v^2(v^3)^t, \ldots, v^2(v^N)^t, v^3(v^2)^t, \ldots, v^3(v^N)^t, \ldots, v^N(v^2)^t, \ldots, v^N(v^N)^t,$$

where $s, v^2, \ldots, v^N$ is an orthonormal basis for $\mathcal{R}^N$.

Proof. Since $G^t = G$ and $G \circ G = G$, G is a projection. The orthonormality follows from the facts that for N-vectors u, v, w, and y,

$$\|uv^t\|_F = |u|\,|v|$$

and

$$(uv^t)^T(wy^t) = (u^t w)(v^t y),$$

where the superscript T represents the inner product in $\mathrm{L}(\mathcal{R}^N, \mathcal{R}^N)$. The eigenvalue properties are easily checked. □

Since $\mathcal{S}$ is an invariant subspace of $G(s; \cdot)$, it is of interest to provide an orthonormal basis for $\mathcal{S}$ consisting of eigenvectors of $G(s; \cdot)$. The following is a direct consequence of Proposition 4.1.

PROPOSITION 4.2. *The following symmetric matrices are orthonormal eigenvectors of $G(s; \cdot)$ corresponding to the eigenvalue zero:*

$$ss^t, \frac{s(v^2)^t + v^2 s^t}{\sqrt{2}}, \ldots, \frac{s(v^N)^t + v^N s^t}{\sqrt{2}};$$

and the following are orthonormal eigenvectors corresponding to the eigenvalue one:

$$v^k(v^k), \frac{v^j(v^k)^t + v^k(v^j)^t}{\sqrt{2}}, \qquad \textit{for} \quad k = 2, \ldots, N \quad \textit{and} \quad j = k+1, \ldots, N.$$

Since $G(s;\cdot)$ has eigenvalues equal to one it is not a contraction mapping, even when restricted to $\mathcal{S}$, and so linear convergence of the E_k is ruled out in general. Since $G(s;\cdot)$ is a projection matrix, it is possible, however, that a sequence of applications will reduce the norm of the E_k, i.e., that some multistep linear convergence can occur. In particular, if we let the sequence of N applications of G starting at s_k be denoted by

$$G_{N,k}(\cdot) = G(s_{k+N-1}, \cdot) \circ G(s_{k+N-2}, \cdot) \circ \cdots \circ G(s_k, \cdot), \tag{4.3}$$

then it is not hard to see that if $s_k, \ldots, s_{k+N-1}$ are linearly independent, then $G_{N,k}$ is a contraction mapping; that is, the operator norm of $G_{N,k}$,

$$\|G_{N,k}\| = \max\{\|G_{N,k}(W)\|_F : W \in L(\mathcal{R}^N, \mathcal{R}^N) \text{ and } \|W\|_F = 1\}$$

is less than one. (In fact, from Proposition 4.2 it is seen that $G_{N,k}(vv^t) = vv^t$ if $v^t s_j = 0$ for $j = k, \ldots, k+N-1$, so that $\|G_{N,k}\| = 1$ if $s_k, \ldots, s_{k+N-1}$ are linearly dependent.) If there are an α, $0 < \alpha < 1$, and an infinite sequence of integers $\{k_j\}$ such that $s_{k_j}, \ldots, s_{k_j+N-1}$ are linearly independent and $\|G_{N,k_j}\| < \alpha$ for all j, then it is clear that the sequence $\{E_k\}$ generated by $\{s_k\}$ and the L update will converge to the zero matrix regardless of the starting matrix E_0.

The above remarks suggest the approach that we shall take for obtaining the convergence of the sequence $\{E_k\}$, not only for the L update, but for others given by (2.5). However, we will considerably relax the requirements on the sequence $\{s_k\}$ and, in addition, will obtain actual rates of convergence for the sequence $\{E_k\}$.

We first extend the notation in (4.3) to cover sequences of operators of length greater than N, i.e., for integers k and$K$, $K \geq N$, let

$$G_{K,k}(\cdot) = G(s_{k+K-1}, \cdot) \circ \cdots \circ G(s_k, \cdot)$$

for a given sequence $\{s_k\}$. Then for any $N \times N$ matrix E,

$$G_{K,k}(E) = R_{K,k}^t E R_{K,k}, \tag{4.4}$$

where

$$R_{K,k} = \prod_{j=k}^{k+K-1} (I - s_j s_j^t).$$

If we let the $N \times K$ matrix with columns $s_k, \ldots, s_{k+K-1}$ be denoted by $S_{K,k}$ and define the $K \times K$ upper triangular matrix $U_{K,k}$ by

$$(U_{K,k})_{i,j} = \begin{cases} s_{k+i-1}^t \quad s_{k+j-1} & i \leq j, \\ 0 & \text{else}, \end{cases}$$

we can analyze $G_{K,k}$ in terms of $R_{K,k}$, $S_{K,k}$, and $U_{K,k}$. (In order to simplify the notation, we will hereafter suppress the subscripts K and k.) It can be shown by induction (see [10]) that

$$R = I - SU^{-1}S^t.$$

Then from the definition of U,

$$U + U^t = S^t S + I$$

and it follows that

$$R^t R = I - SU^{-t}U^{-1}S^t. \tag{4.5}$$

The definition of the operator norm of G and (4.4) now imply

$$\|G\| \leq \|R\|^2 = \|R^t R\|.$$

This leads to the following bound on $\|G\|$.

PROPOSITION 4.3 ([15]). *Let K and k be fixed and let S have full row rank N. Then*

$$\|G\| \leq 1 - \frac{1}{K^2}(\|(SS^t)^{-1}\|)^{-1}. \tag{4.6}$$

Proof. We have from (4.5) and the definition of the operator norm

$$\begin{aligned}
\|G\| &\leq 1 - \min_{|u|=1}\{u^t SU^{-t}U^{-1}S^t u\} \\
&= 1 - \min_{|u|=1}\{|U^{-1}S^t u|^2\} \\
&= 1 - \min_{|u|=1}\{|z|^2 : Uz = S^t u\} \\
&\leq 1 - \min_{|u|=1}\left\{\frac{|S^t u|^2}{\|U\|^2}\right\} \\
&= 1 - \frac{1}{\|U\|^2}\min_{|u|=1}\{u^t SS^t u\} \\
&= 1 - \frac{1}{\|U\|^2}(\|(SS^t)^{-1}\|)^{-1}
\end{aligned}$$

and the result follows from the inequality $\|U\| \leq K$. □

Proposition 4.3 motivates the following definition.

DEFINITION 4.1. Let $\{s_k\}$ be a sequence of N-dimensional unit vectors. Let $K \geq N$ be a fixed integer. If there exist an infinite sequence of integers $\{k_j\}$ and a positive constant γ such that the matrices $S_{K,k_j}S_{K,k_j}^t$ are nonsingular and satisfy

$$\|(S_{K,k_j}S_{K,k_j}^t)^{-1}\| \leq \gamma$$

for all j, $\{s_k\}$ is said to be *K-step subsequentially linearly independent.* In this case, if the subsequence $\{k_j\}$ can be chosen so that

$$\chi = \lim_{j\to\infty} \sup\{k_{j+1} - k_j - 1\}$$

is finite, $\{s_k\}$ is said to be *K-step uniformly linearly independent with gap* χ.

Note that the matrices $(S_{K,k_j}S_{K,k_j}^t)^{-1}$ satisfy a bound of the type above if the set of vectors $s_{k_j}, \ldots, s_{k_j+K-1}$ contains a subset of vectors that spans $\mathcal{R}^N$ and is not "nearly" linearly dependent. Since K can be much greater than N and the sets of vectors containing the spanning subsets can be arbitrarily far apart, sequences that are subsequentially linearly independent are quite general. The convergence results that will follow may fail without the

assumption of this bound as it can be shown that $\|G_{K,k_j}\|$ can approach 1 as the set of vectors $s_{k_j}, \ldots, s_{k_j+K-1}$ becomes more nearly linearly dependent, i.e., $S_{K,k_j} S_{K,k_j}^t$ becomes more ill conditioned. The slightly stronger assumption of a finite gap will be required to establish convergence rates.

Using this definition we can now obtain the basic convergence theorem. We first put the system (4.2) into the form of the system (3.1) by identifying each E_k with an N^2-dimensional vector z_k and each $G(s_k;\cdot)$ with an $N^2 \times N^2$ matrix A_k. (Z_k and W_k are both zero for all k.) Since we are employing the Frobenius norm on the E_k, we have $|z_k| = \|E_k\|_F$ and also $\|A_k\| = \|G(s_k;\cdot)\|$. The following proposition will help establish the rate of convergence of the $\{E_k\}$.

PROPOSITION 4.4. *Let the sequence $\{s_k\}$ be given and for each k let A_k be the $N^2 \times N^2$ matrix representing $G(s_k;\cdot)$. Let $\{\beta_{k,m}\}$ be the constants defined by* (3.2). *Suppose $\{s_k\}$ is K-step subsequentially linearly independent and let the constant γ and the subsequence $\{k_j\}$ be as specified in Definition* 4.1. *Then*

$$\beta_{k,1} \leq 1 \tag{4.7}$$

for all k, and

$$\beta_{k_j,K} \leq \alpha = \left(1 - \frac{1}{K^2 \cdot \gamma}\right) < 1 \tag{4.8}$$

for each j. If $\{s_k\}$ is K-step uniformly linearly independent with gap χ then

$$\beta_{k,(K+\chi)} \leq \alpha = \left(1 - \frac{1}{K^2 \cdot \gamma}\right) < 1 \tag{4.9}$$

for all k sufficiently large.

Proof. Equations (4.7) and (4.8) follow directly form Definition 4.1. Assuming $\{s_k\}$ is K-step uniformly linearly independent with gap χ, let $\{k_j\}$ be the sequence given by Definition 4.1. Then for every k sufficiently large, there is a k_j such that $k_j - \chi \leq k \leq k_j$. Therefore from (4.6) and Definition 4.1

$$\begin{aligned}\left\| \prod_{i=0}^{K+\chi-1} A_{k+i} \right\| &\leq \left(\prod_{i=k-k_j}^{-1} \|A_{k_j+i}\| \right) \cdot \left\| \left(\prod_{i=0}^{K-1} A_{k_j+i} \right) \right\| \cdot \left(\prod_{i=0}^{k-k_j+\chi-1} \|A_{k_j+K+i}\| \right) \\ &\leq \left(1 - \frac{1}{K^2 \cdot \gamma}\right) < 1\end{aligned}$$

which yields (4.9). □

The main result of this section is now just a straightforward application of Theorem 3.1 (with Z_k and W_k both zero) and Proposition 4.4.

THEOREM 4.5. *Suppose the sequence $\{s_k\}$ is K-step subsequentially linearly independent. Then the sequence $\{E_k\}$ defined by* (4.1) *converges to the zero matrix from any symmetric starting matrix E_0. If the sequence $\{s_k\}$ is K-step uniformly linearly independent with gap χ, then the convergence rate is $(K+\chi)$-step linear.*

Theorem 4.5 yields the convergence of the sequence $\{E_k\}$ generated by the L-update under the relatively weak conditions on the sequence $\{s_k\}$. As was pointed out in §1, sequences $\{s_k\}$ can be constructed for which the corresponding $\{E_k\}$ will not converge. This was demonstrated by Ge and Powell [4] for the two-dimensional case. In their example, the vectors $\{s_k\}$ are pairwise linearly independent but become more nearly identical while not converging

to a single vector. Thus the vectors are not K-step subsequentially linearly independent for any K, and they do not converge to a proper subspace of $\mathcal{R}^2$. This latter property is also necessary for nonconvergence of the $\{E_k\}$ in a sense that will be made clear in the following section.

5. Extensions of convergence results for the L update. Theorem 4.5 is predicated on the assumption of the (subsequential) uniform linear independence of the sequence $\{s_k\}$, which requires that the vectors s_k repeatedly span $\mathcal{R}^N$. We now relax this restriction and assume instead that they span a proper M-dimensional subspace, S, of $\mathcal{R}^N$. To analyze this case we let P be an $N \times M$ matrix whose columns are an orthonormal basis for S and let Q be an $N \times (N-M)$ matrix whose columns are an orthonormal basis for $S^{\perp}$. For such P and Q any $N \times N$ symmetric E can be written

$$E = PP^tEPP^t + QQ^tEQQ^t + PP^tEQQ^t + QQ^tEPP^t. \tag{5.1}$$

Note that PP^t is the projection of $\mathcal{R}^N$ onto S and $QQ^t = 1 - PP^t$ is the projection onto $S^{\perp}$. For each k we let $r_k = P^ts_k$ or, equivalently, $s_k = Pr_k$, where $r_k \in \mathcal{R}^M$ and $|r_k| = 1$. For a sequence $\{E_k\}$ satisfying (4.1), we set

$$V_k = P^tE_kP, \tag{5.2a}$$

$$U_k = Q^tE_kQ, \tag{5.2b}$$

$$Y_k = P^tE_kQ, \tag{5.2c}$$

so that

$$E_k = PV_kP^t + QU_kQ^t + PY_kQ^t + QY_k^tP^t. \tag{5.3}$$

Note that $Y_k = 0$ if and only if S is an eigenspace of E_k. Since $P^tP = I_M$ (the $M \times M$ identity matrix) and $Q^ts_k = 0$, we see from (4.1) that these matrices satisfy the following uncoupled difference equations:

$$V_{k+1} = (I_M - r_kr_k^t)\, V_k(I_M - r_kr_k^t), \tag{5.4a}$$

$$U_{k+1} = U_k, \tag{5.4b}$$

$$Y_{k+1} = (I_M - r_kr_k^t)Y_k. \tag{5.4c}$$

Equation (5.4a) shows that the $M \times M$ matrices, V_k, satisfy the updating scheme (4.1); (5.4b) shows that the $(N-M) \times (N-M)$ matrices, U_k, are constant with respect to k; and (5.4c) shows that the $M \times (N-M)$ matrices, Y_k, satisfy a one-sided type of (4.1) update. Thus if the sequence $\{r_k\}$ is K-step uniformly linearly independent with gap χ, it follows from Theorem 4.5 that $\{V_k\}$ converges to the $M \times M$ zero matrix and that the rate of convergence is $(K+\chi)$-step linear. A modification of the proofs of the propositions and theorems in §4 can be used to show that the Y_κ also converge to a zero matrix at the $(K+\chi)$-linear rate. It is perhaps easier, however, to note that the symmetric positive semidefinite matrices $Z_k = Y_kY_k^t$ satisfy the same equation as the V_k and hence tend to zero; the convergence of the Y_k to the zero matrix R-linearly then follows from the convergence of the Z_k. If the sequence $\{r_k\}$ is only K-step subsequentially linearly independent, then the convergence of the V_k and Y_k still holds but there is no multistep linear rate. As a result of these observations we have the following theorem that shows that the restriction of the $\{s_k\}$ to a subspace does not prevent the convergence of the $\{E_k\}$.

THEOREM 5.1. *Let S be an M-dimensional subspace of $\mathcal{R}^N$ and let the matrices P and Q be defined as above. Suppose the sequence of N-vectors $\{s_k\}$ is contained in S and that the sequence of M-vectors, $\{r_k\}$, defined by $r_k = P^t s_k$ is K-step subsequentially linearly independent. Then for any initial symmetric matrix E_0, the sequence defined by* (4.1) *converges to the matrix $QQ^tE_0QQ^t$. If the sequence $\{r_k\}$ is K-step uniformly linearly independent with gap χ then the convergence rate is at least* R-*linear.*

Proof. By the remarks above, $V_k \to 0$, $Y_k \to 0$, and $U_k = U_0$ for all k. Thus the convergence result follows from (5.3). The R-linear convergence follows from the fact that the components of E_k have $(K+\chi)$-step linear and hence R-linear convergence, which implies (by the remarks in §3) the R-linear convergence of the E_k. □

We note that the proof shows that the E_k converge $(K + \chi)$-step linearly in the norm defined by

$$\| |E| \| = \|PP^tEPP^t\|_F + \|QQ^tEQQ^t\|_F + \|QQ^tEPP^t\|_F + \|PP^tEQQ^t\|_F.$$

This will also be the case in some, but not all, of the results that follow. We do not pursue this property since we do not feel it yields a significant improvement over R-linear convergence.

The next result is somewhat surprising in that it demonstrates that if $\{s_k\}$ converges at a reasonable rate to a subspace, convergence of the $\{E_k\}$ can still be obtained. The importance of this type of convergence in constrained optimization will be discussed in §8.

DEFINITION 5.1. Let S be an M-dimensional subspace of $\mathcal{R}^N$ and suppose that $\{s_k\}$ is a sequence of unit vectors that satisfy $s_k = u_k + v_k$ where $u_k \in S$, $v_k \in S^\perp$, and $v_k \to 0$ as $k \to \infty$. Then we say that $\{s_k\}$ *converges to* S. If $|v_k|$ tends to zero R-linearly, we say that $\{s_k\}$ converges to S at an R-linear rate.

THEOREM 5.2. *Suppose that the $\{s_k\}$ converge to an M-dimensional subspace S at an* R-*linear rate. Let matrices P and Q be defined as above, let $\{r_k\}$ be the sequence of unit M-vectors given by*

$$r_k = \frac{P^t s_k}{|P^t s_k|}, \tag{5.5}$$

and assume that $\{r_k\}$ is K-step subsequentially linearly independent. Then for any initial matrix E_0 the sequence $\{E_k\}$ defined by (4.1) *converges to $QQ^tE^*QQ^t$ for some $N \times N$ symmetric matrix E^*. If $\{r_k\}$ is K-step uniformly linearly independent with gap χ, then the convergence rate is* R-*linear.*

Proof. Let $s_k = u_k + v_k$ as in Definition 5.1. Then

$$s_k = \frac{u_k}{|u_k|} + \alpha_k u_k + v_k,$$

where $\alpha_k = (|u_k| - 1)/|u_k|$, and from (5.5)

$$r_k = \frac{P^t u_k}{|u_k|},$$

and hence

$$s_k = Pr_k + \alpha_k u_k + v_k.$$

It can be shown that the conditions $|s_k| = 1$ and $|v_k| \to 0$ R-linearly imply that $\alpha_k \to 0$ R-linearly. Thus we can write

$$s_k = Pr_k + t_k, \tag{5.6}$$

where $t_k \to 0$ R-linearly. Now using (5.1) and letting V_k, U_k, and Y_k be defined as in (5.2), we obtain the equations

$$V_{k+1} = (I - r_k r_k^t) V_k (I - r_k r_k^t) + \mathcal{V}(E_k, r_k, t_k), \tag{5.7a}$$

$$U_{k+1} = U_k + \mathcal{U}(E_k, r_k, t_k), \tag{5.7b}$$

$$Y_{k+1} Y_{k+1}^t = (I - r_k r_k^t) Y_k Y_k^t (I - r_k r_k^t) + \mathcal{Y}(E_k, r_k, t_k), \tag{5.7c}$$

where, since the E_k and r_k are uniformly bounded, the nonlinear terms $\mathcal{V}$, $\mathcal{U}$, and $\mathcal{Y}$ are $O(|t_k|)$. Thus the system (5.7) can be put in the framework of (3.1) by identifying V_k and $Y_k Y_k^t$ with z_k, and U_k with w_k. Then from the conclusion of Theorem 3.1 $\{U_k\}$ converges to some symmetric matrix U^*, which in turn implies that $\{QU_kQ^t\}$ converges to $QU^*Q^t = E^*$. Then we have $U^* = Q^t E^* Q$ and hence $\{E_k\}$ converges to $QQ^t E^* QQ^t$. □

As a result of this theorem, it is seen that nonconvergence of $\{E_k\}$ can occur only if the sequence $\{s_k\}$ does not approach any fixed subspace R-linearly in such a way that the projections onto the subspace are K-step subsequentially linearly independent for any K. As noted in §4, in the Ge and Powell example [4] the vectors approach each other but not a fixed subspace.

6. Convergence results for other updates. In this section we extend the convergence results obtained in §4 and 5 to the update process (2.5) for ϕ_k other than zero. This will be accomplished by considering the general update as a perturbation of the L update. That is, from (2.5)

$$E_{k+1} = G(s_k; E_k) - \phi_k H(s_k; E_k) \tag{6.1}$$

where $H : L(\mathcal{R}^N, \mathcal{R}^N) \to L(\mathcal{R}^N, \mathcal{R}^N)$ is a nonlinear mapping. If $\|E_k\|_F$ is sufficiently small then

$$|(1 + s_k^t E_k s_k)^{-1}| = 1 + O(\|E_k\|_F),$$

and hence

$$\|H(s_k; E_k)\|_F \leq \alpha (\|E_k\|_F)^2 \tag{6.2}$$

where α is a constant independent of k. This fact leads directly to the next theorem.

THEOREM 6.1. *Let $\{s_k\}$ be K-step uniformly linearly independent with gap χ and let the ϕ_k satisfy*

$$|\phi_k| \leq \kappa \cdot (1 + \|E_k\|_F) \tag{6.3}$$

for some constant κ independent of k and $\|E_k\|_F$ sufficiently small. Then there exists a positive constant δ such that if E_0 is any symmetric matrix satisfying $\|E_0\|_F < \delta$ the sequence defined by (2.5) *converges to the zero matrix and the rate of convergence is $(K + \chi)$-step linear.*

Proof. As in the proof of Theorem 4.5 we identify E_k with z_k and $G(s_k; \cdot)$ with A_k. Now identifying the term $\phi_k H(s_k; \cdot)$ with Z_k in (3.1) and applying (6.2) and (6.3) assures the hypotheses of Theorem 3.2 are satisfied. (No w_k is present in this application.) The result follows. □

The boundedness requirement on the ϕ_k, (6.3), generally rules out dynamic processes such as the SR1 in which the ϕ_k can be arbitrarily large for small E_k (although the SR1 update itself will converge finitely in this case). There are two significant differences between Theorem 6.1 and the corresponding theorem for the L update, Theorem 4.5. First, it appears that K-step uniform linear independence is needed as a hypothesis for Theorem 6.1; otherwise,

the gap between successive values of the sequence $\{k_j\}$ in Definition 4.2 could become large enough to allow the quadratic terms in $H(s_k;\cdot)$ to grow so rapidly as to preclude convergence of the solution of (3.1) to zero. More importantly, Theorem 6.1 yields local convergence only; inequality (6.2) is not valid without the assumption that $\|E_0\|$ is small. However, as will be observed in §7, for certain starting matrices the convergence of the C updating process can be obtained without these two restrictions.

To obtain the convergence of the general updating scheme (2.5) when the $\{s_k\}$ lie on a subspace, we apply the perturbation technique of Theorem 6.1 to the analysis in Theorem 5.1. Defining the matrices P, Q, V_k, U_k, and Y_k as in §5 and setting $r_k = P^t s_k$ permit us to decompose (2.5) into the system

$$\begin{aligned} V_{k+1} &= (I - r_k r_k^t)V_k(I - r_k r_k^t) \\ &\quad - \phi_k \left[\frac{1}{1 + r_k^t V_k r_k}\right] [(r_k^t V_k r_k)r_k - V_k r_k][(r_k^t V_k r_k)r_k - V_k r_k]^t, \end{aligned} \tag{6.4a}$$

$$Y_{k+1} = (I - r_k r_k^t)Y_k - \phi_k \left[\frac{1}{1 + r_k^t V_k r_k}\right] [(r_k^t V_k r_k)r_k - V_k r_k][r_k^t Y_k], \tag{6.4b}$$

$$U_{k+1} = U_k - \phi_k \left[\frac{1}{1 + r_k^t V_k r_k}\right] Y_k^t r_k r_k^t Y_k. \tag{6.4c}$$

From these equations we obtain the following result.

THEOREM 6.2. *Let S be an M-dimensional subspace of $\mathcal{R}^N$ and let the matrices P, Q, V_k, Y_k, and U_k be defined as in §5. Suppose the sequence of N-vectors $\{s_k\}$ is contained in S and that the sequence of M-vectors, $\{r_k\}$, defined by $r_k = P^t s_k$ is K-step uniformly linearly independent with gap χ. In addition suppose that the ϕ_k satisfy*

$$|\phi_k| \leq \kappa \cdot (1 + \|P^t E_k P\|_F) \tag{6.5}$$

for $\|P^t E_k P\|_F$ sufficiently small, where κ is independent of k. Then there exists a positive constant δ such that for any initial symmetric matrix E_0 satisfying $\|P^t E_0 P\|_F \leq \delta$ the sequence defined by (2.5) converges to a matrix of the form $QQ^t E^* QQ^t$ for some $N \times N$ symmetric matrix E^* and the convergence rate is at least R*-linear.*

Proof. The equation (6.4a) for V_k is identical in form to that of (2.5). Therefore, by Theorem 6.1 $V_k \to 0$ at an $(K + \chi)$-step rate and hence an R-linear rate. Then by identifying V_k with t_k, $Y_k Y_k^t$ with z_k, and U_k with w_k, it is seen that the last two equations in (6.4) have the form of the system (3.1) and satisfy the hypotheses of Theorem 3.2. The result follows as in the proof of Theorem 5.2. □

The matrix E^* is not specified by the theorem. In particular it is not necessarily E_0 as in Theorem 5.1. It should also be observed that the convergence depends only on the initial value of the $PP^t E_0 PP^t$ component of E_0.

If the s_k are not on the subspace S but converge to it in a R-linear manner, then by using the combination of the analysis of Theorem 5.2 with that of the preceding theorem the following result can be obtained.

THEOREM 6.3. *Let the subspace S be given and let the matrices P and Q be defined as above. For each k let $s_k = Pr_k + t_k$ where $\{r_k\}$ is the sequence of M-vectors defined by* (5.5) *and t_k is defined by* (5.6). *Assume that $\{r_k\}$ is K-step uniformly linearly independent with gap χ and that*

$$|t_k| \leq |t_0|\tau^k \tag{6.6}$$

for some constant τ, $0 < \tau < 1$. *Also assume that the sequence* $\{\phi_k\}$ *satisfies*

$$|\phi_k| \le \kappa(1 + \|P^t E_k P\|_F + |t_k| \cdot \|E_k\|_F) \tag{6.7}$$

for some constant κ *independent of* k. *Let* E_0 *be a given symmetric matrix. Then there exist positive constants* $\tau_0 = \tau(E_0)$ *and* δ *such that if* $\|P^t E_0 P\|_F + \|P^t E_0 Q\|_F < \delta$ *and* $|t_0| \le \tau_0$, *the sequence* $\{E_k\}$ *defined by* (2.5) *converges* R-*linearly to a matrix of the form* $QQ^t E^* QQ^t$.

Proof. (6.6) implies that $t_k \to 0$ at an R-linear rate (and hence $\{s_k\}$ converges to S at an R-linear rate). Using (5.2) we obtain, from (2.5),

$$\begin{aligned}
V_{k+1} &= (I - r_k r_k^t) V_k (I - r_k r_k^t) \\
&\quad - \phi_k \left[\frac{1}{1 + r_k^t V_k r_k}\right] [(r_k^t V_k r_k) r_k - V_k r_k][(r_k^t V_k r_k) r_k - V_k r_k]^t \\
&\quad + \mathcal{V}_k(r_k, V_k, U_k, Y_k, t_k), \\
Y_{k+1} &= (I - r_k r_k^t) Y_k - \phi_k \left[\frac{1}{1 + r_k^t V_k r_k}\right] [(r_k^t V_k r_k) r_k - V_k r_k][r_k^t Y_k] \\
&\quad + \mathcal{Y}_k(r_k, V_k, U_k, Y_k, t_k), \\
U_{k+1} &= U_k - \phi_k \left[\frac{1}{1 + r_k^t V_k r_k}\right] Y_k^t r_k r_k^t + \mathcal{U}_k(r_k, V_k, U_k, Y_k, t_k).
\end{aligned}$$

Since the r_k are uniformly bounded, $\mathcal{V}_k$, $\mathcal{U}_k$, and $\mathcal{Y}_k$ are $O((\|V_k\|_F + \|U_k\|_F + \|Y_k\|_F) \cdot |t_k|)$. Using (6.7) and identifying V_k and $Y_k Y_k^t$ with z_k and U_k with w_k in (3.1), we see that the hypotheses of Theorem 3.2 are satisfied and hence the results follow. □

The hypotheses of this theorem do not require that the component of E_0 relative to $S^\perp$ be small; however, they do require that the components of $\{s_k\}$ start and remain close to S. An analysis of the proof of Theorem 3.2 will show that if the component of E_0 relative to $S^\perp$ is sufficiently small then the requirement that the $\{s_k\}$ start close to S can be relaxed, although the sequence still must converge to S at an R-linear rate.

7. Application to updates of H_k**.** In the previous three sections we have established the convergence of the sequence of matrices $\{E_k\}$ generated by the system (2.5) under certain restrictions on the sequences $\{s_k\}$ and $\{\phi_k\}$. We now apply those results to the sequence $\{H_k\}$ generated by (1.5). While the results obtained so far may be of interest in their own right, the updating schemes (1.5) are of practical interest in the optimization algorithms that motivated this study and are, consequently, of more import. For the simplest (quadratic) case, i.e., when $y_k = s_k$, the translation of the major theorems of §§4–6 into results for the system (2.2) can be carried out by a straightforward substitution of $H_k - I$ for E_k. The results of this transformation are given in the next two theorems in condensed form; the first theorem deals with the case where the sequence $\{s_k\}$ remains on the subspace S (which may be $\mathcal{R}^N$) and the second with the case where $\{s_k\}$ converges to a proper subspace S.

THEOREM 7.1. *Let* S *be an* M*-dimensional subspace of* $\mathcal{R}^N$ *with* $M \le N$ *and let the matrices* P *and* Q *be defined relative to* S *as in* §5 ($P = I$ *and* $Q = 0$ *if* $M = N$). *Suppose each vector in the sequence of unit* N*-vectors,* $\{s_k\}$, *is in* S *and suppose that the sequence* $\{r_k\}$, *defined by* $r_k = P^t s_k$, *is* K*-step uniformly independent with gap* χ *in* S. *Let* H_0 *be a symmetric matrix and generate the sequence* $\{H_k\}$ *by* (2.2). *It follows that*

(i) *If* $\phi_k = 0$ *for each* k, *then*

$$\lim_{k \to \infty} H_k = QQ^t H_0 QQ^t + PP^t;$$

or

(ii) *If the* ϕ_k *satisfy* $|\phi_k| \leq \kappa(1 + \|P^t H_k P\|_F)$ *for some* κ *independent of* k *and* $\|P^t H_0 P - I_M\|_F$ *is sufficiently small, then for some* $N \times N$ *symmetric matrix* H^*

$$\lim_{k\to\infty} H_k = QQ^t H^* QQ^t + PP^t.$$

The convergence rate of the matrices is $(K + \chi)$-step linear if $S = \mathcal{R}^N$ and R-linear if S is a proper subspace of $\mathcal{R}^N$. The K-step uniform linear independence can be replaced in case (i) by K-step subsequential linear independence, but the R-linear rate convergence is no longer guaranteed.

THEOREM 7.2. *Let* S *be a subspace of* $\mathcal{R}^N$ *and assume that the sequence* $\{s_k\}$ *satisfies the hypotheses of Theorem* 6.3. *Assume that the* ϕ_k *satisfy*

$$|\phi_k| \leq \kappa(1 + \|P^t H_k P\|_F + |t_k| \cdot \|H_k\|_F)$$

for some κ *independent of* k. *Let* H_0 *be a given symmetric matrix. Then there exist constants* $\tau_0 = \tau(H_0)$ *and* δ *such that if* $\|P^t H_0 P - I_M\|_F + \|P^t H_0 Q\|_F \leq \delta$ *and* $\tau \leq \tau_0$, *then the sequence* $\{H_k\}$ *generated by* (1.5) *satisfies*

$$\lim_{k\to\infty} H_k = QQ^t H^* QQ^t + PP^t$$

for some $N \times N$ *symmetric matrix* H^* *and the convergence rate is* R-*linear. If* $\phi_k = 0$ *for all* k, *then the restriction on the initial matrix* H_0 *is not required.*

Thus, generally speaking, the sequence of updates defined by (2.2) converges to a matrix that is the identity on the subspace spanned by the sequence $\{s_k\}$ (possibly in a limiting sense) provided the initial matrix is nearly the projection PP^t. In particular, the subspaces S and $S^\perp$ are eigenspaces of the limiting matrix.

We now drop the assumptions that each s_k is a unit vector and that $y_k = s_k$ and require instead that at each iteration

$$y_k = s_k + \xi_k, \tag{7.1}$$

where $\xi_k/|s_k| \to 0$ as $k \to \infty$. As discussed in §2 this assumption is motivated by the unconstrained optimization problem where s_k represents the difference in successive iterates. We then have the following theorem.

THEOREM 7.3. *Let* $\{s_k\}$ *satisfy the hypotheses of Theorem* 6.3. *Let the sequence* $\{y_k\}$ *satisfy* (7.1) *and assume that* $\xi_k/|s_k| \to 0$ *at an* R-*linear rate, i.e.,* $|\xi_k|/|s_k| \leq \eta_0 \cdot \eta^k$ *for all* k *and some* η_0, *where* $0 \leq \eta < 1$. *Assume that the sequence* $\{\phi_k\}$ *satisfies*

$$|\phi_k| \leq \kappa(1 + \|P^t H_k P\|_F + (|t_k| + |\xi_k|/|s_k|) \cdot \|H_k\|_F)$$

for some constant κ *independent of* k. *Let* H_0 *be a symmetric matrix. Then there exist positive constants* δ *and* $\tau_0 = \tau(H_0)$ *such that for* $\|P^t H_0 P - I_M\|_F + \|P^t H_0 Q\|_F < \delta, \eta_0 < \tau_0$, *and* $|t_0| < \tau_0$, *the sequence* $\{H_k\}$ *generated by* (1.5), *satisfies*

$$\lim_{k\to\infty} H_k = QQ^t H^* QQ^t + PP^t$$

for some $N \times N$ *symmetric matrix* H^* *and the convergence rate is* R-*linear. If* $\phi_k = 0$ *for all* k *then no restriction is required on the initial matrix* H_0.

Proof. Let (7.1) hold. Then

$$y_k^t s_k = (s_k^t s_k)\left(1 + \frac{s_k^t \xi_k}{s_k^t s_k}\right)$$

and hence if $|\xi_k|/|s_k|$ is sufficiently small,

$$(y_k^t s_k)^{-1} = (s_k^t s_k)^{-1}(1 + O(|\xi_k|/|s_k|)). \tag{7.2}$$

Substituting (7.1) and (7.2) into the update formula (1.5), making the changes of variables

$$\hat{s}_k = \frac{s_k}{|s_k|} \quad \text{and} \quad \hat{\xi}_k = \frac{\xi_k}{|s_k|},$$

and replacing, as before, H_k by $E_k + I$ yield an equation of the form

$$E_{k+1} = G(\hat{s}_k; E_k) - \phi_k H(\hat{s}_k; E_k) + \mathcal{W}(\phi_k, \hat{s}_k, E_k, \hat{\xi}_k), \tag{7.3}$$

where now $|\hat{s}_k| = 1$, and $|\hat{\xi}_k| \leq |\eta_0| \cdot \eta^k$. Using the bound on the ϕ_k, it is seen that

$$\|\mathcal{W}(\phi_k, \hat{s}_k, E_k, \hat{\xi}_k)\|_F \leq \kappa(\|E_k\|_F) \cdot |\hat{\xi}_k|. \tag{7.4}$$

Now the $\hat{s}_k$ can be decomposed as in Theorem 6.3 and the results of §3 (in particular Theorem 3.2) can be applied just as before. □

It may be disconcerting to observe that Theorems 7.1–7.3 yield global convergence (i.e., unrestricted initial matrix) of $\{H_k\}$ only in the case where ϕ_i is zero. Proposition 2.2 allows us to extend these global convergence results to the C updates for certain cases, e.g., those where the L updates starting at H_0^{-1} is guaranteed to converge to a nonsingular matrix.

THEOREM 7.4. *Suppose that $y_k = s_k \in S$ for every k and that either*

(i) $S = \mathcal{R}^N$ and H_0 *is nonsingular,*

or

(ii) H_0 *is positive definite.*

Then if $\{s_k\}$ is K-step uniformly linearly independent in S and $\phi_k = 1$ for all k, the sequence $\{H_k\}$ converges to

$$Q(Q^t H_0^{-1} Q)^{-1} Q^t + PP^t.$$

Proof. The theorem follows directly from Theorem 7.1, part (i). The iteration process with $\phi_k = 0$ for each k applied to H_0^{-1} converges to $QQ^t H_0^{-1} QQ^t + PP^t$. This limit is nonsingular in both cases and hence by Proposition 2.2, the iterates using $\phi_k = 0$ and starting at H_0 converge to

$$[QQ^t H_0^{-1} QQ^t + PP^t]^{-1}.$$

An easy calculation now yields the result. □

It would be ideal to extend Theorem 7.4 to the cases where $\{s_k\}$ satisfies the hypotheses of Theorems 7.2 and 7.3. However, the argument above fails since the limit, of the form $QQ^t H^* QQ^t + PP^t$, may be singular for some H^*.

In addition to the class of "direct" Broyden updates generated by (1.5), there is the class of "inverse" Broyden updates that are obtained from (1.5) by interchanging s and y. In the unconstrained optimization setting this class of updates generates sequences $\{H_k\}$ that represent approximations to the inverse of the Hessian matrix of the objective function. When the $y_k = s_k$, the resulting class reduces to (2.2) and the theorems of §§4–6 apply. When (7.1) holds, Theorem 7.3 can be repeated for the "inverse" class with no essential change. Only the function $\mathcal{W}$ is slightly different, although it still satisfies (7.4).

When the theorems above are applied to the unconstrained optimization problem, it is under the assumption that the Hessian matrix of the objective function at the optimal solution

is the identity. Since the form of the Broyden class of updates (1.5), as well as the form of the "inverse" class, is invariant under the transformations (T) in §2, there is no loss in generality in making this assumption. The limiting value of the Hessian approximations in the untransformed case will be the Hessian matrix F if the vectors $F^{1/2}s_k$, equivalently the vectors s_k, do not approach a proper subspace of $\mathcal{R}^N$. If $\{F^{1/2}s_k\}$ approaches a subspace S which is an eigenspace of F, the matrices P and Q can be chosen so that the columns are eigenvectors of F (and, hence, $F^{1/2}$). Then

$$F^{1/2}[P,Q] = [P,Q]D = [P,Q]\begin{bmatrix} D_1 & 0 \\ 0 & D_2 \end{bmatrix},$$

where D_1 and D_2 are the diagonal matrices whose diagonal elements are the square roots of the eigenvalues of F on the subspaces S and $S^{\perp}$, respectively. Now the general form for the limiting matrix gives

$$\lim_{k\to\infty} F^{-1/2}H_k f^{-1/2} = PP^t + QQ^t H^* QQ^t;$$

so for $U^* = Q^t H^* Q$,

$$\begin{aligned}\lim_{k\to\infty} H_k &= F^{1/2}[P,Q]\begin{bmatrix} I & 0 \\ 0 & U^* \end{bmatrix}[P,Q]^t F^{1/2}, \\ &= [P,Q]D\begin{bmatrix} I & 0 \\ 0 & U^* \end{bmatrix}D[P,Q]^t, \\ &= PD_1^2P^t + Q\hat{U}Q^t\end{aligned}$$

for $\hat{U} = D_2Q^tH^*QD_2$. If S is not an eigenspace of F, then the columns of P and Q cannot be chosen as eigenvectors of F and the limit is $F^{1/2}[PP^t + QQ^tH^*QQ^t]F^{1/2}$.

Finally, it should be pointed out that other rank-two updates not of the Broyden class may be able to be analyzed by the methods presented here. For example, consider the well-known PSB update which has the form

$$\mathcal{U}(H,s,y) = H + \frac{(y-Hs)s^t + s(y-Hs)^t}{s^ts} - \frac{(y-Hs)^t}{(s^ts)^2}ss^t.$$

This update is not invariant under the transformations given above; however, we can analyze it by letting $y = Fs + \xi$ where $\xi \to 0$ as above. Then letting $E = H - F$ and $\xi = 0$, we obtain the L updating scheme (4.1). Thus we can prove, under the appropriate restrictions on the $\{s_k\}$, that the sequence $\{H_k\}$ converges to F if $\{s_k\}$ does not converge to a subspace. If $\{s_k\}$ converges to an eigenspace, S, of F, then we obtain

$$\lim_{k\to\infty}(H_k - F) = QQ^t\hat{H}QQ^t$$

for some matrix $\hat{H}$. Now since the limiting subspace of $\{s_k\}$ is an eigenspace of F,

$$F = PD_1P^t + QD_2Q^t,$$

where D_1 and D_2 are diagonal matrices. (As above, P and Q are chosen to have eigenvectors of F as columns.) Thus

$$\lim_{k\to\infty} H_k = PD_1P^t + Q\hat{U}Q^t$$

for some $(N - M) \times (N - M)$ matrix $\hat{U}$. This is a somewhat more general result than is available for the Broyden class of updates because there is no restriction that the limiting matrix F be positive definite; it only need be symmetric. In the Broyden class of updates, the term $y^t s$ that appears in the denominator can be zero when $y = Fs$ and F is not positive definite. Therefore, unless the s_k lie on, or converge to, a subspace on which F is positive definite, convergence results cannot be obtained for the Broyden class of updates when F is indefinite. The implications of these remarks for constrained optimization algorithms are discussed in the next section.

8. Final comments. We have analyzed the convergence of a sequence of symmetric matrices $\{H_k\}$, satisfying the rank-two updating formulas (1.5) of the type typically encountered in optimization algorithms. This analysis was carried out independently of the particular algorithm and, indeed, independently of any optimization problem. The results indicate that the convergence of the matrices follows from the secant condition (1.2), the requirement that the sequence $\{s_k\}$ satisfy certain linear independence conditions, and the condition that the sequence $\{y_k - s_k\}$ approach zero at a specified rate. It does not postulate that the sequence $\{s_k\}$ be related to any optimization iteration scheme. In addition to establishing convergence of the sequence of matrices the results presented show that the limiting matrix is the identity matrix when restricted to the subspace to which $\{s_k\}$ converges. In optimization problems where $y_k - F(x^*)s_k$ approaches zero and the Hessian matrix $F(x^*)$ is positive definite with the subspace as an eigenspace, the limit of the matrices is identical to $F(x^*)$ on the subspace.

These results do not subsume those of earlier research, e.g., those contained in [3]–[7], because the conditions on the sequences $\{s_k\}$ and $\{y_k\}$ are assumed here and not derived as a consequence of their generation in optimization problems. They do, however, provide characterizations of the limiting matrix not given in those works and, in particular, illustrate how its structure depends upon the subspace that is spanned by the sequence $\{s_k\}$, an important consideration in, for example, constrained optimization. Another distinction of the results presented here is that some analysis of convergence rates is given. The convergence rate of these matrices, under the assumptions used, is essentially R-linear; thus it is unlikely to be recognized in an algorithm where the x-iterates are converging much more rapidly, such as in a Q-superlinearly convergent algorithm.

One of the original motives for undertaking this study was to try to find alternative proofs of the superlinear convergence of the BFGS and DFP algorithms for unconstrained optimization. In particular, we wanted to try to simplify the proofs' dependence on the bounded deterioration estimates and the use of different weighted norms [1]. The relation between the superlinear convergence of these algorithms and the convergence of the Hessian approximations is best understood by recalling that if the iterates $\{x_k\}$ converge and $x_{k+1} - x_k = s_k$, then the iterates converge superlinearly if and only if

$$\lim_{k \to \infty} (H_k - I)\frac{s_k}{|s_k|} = 0. \tag{8.1}$$

(We assume the Hessian is the identity at the solution.) Now if $\{s_k\}$ converges to a subspace S R-linearly and the projections of the s_k span a subspace S in the K-step uniformly linearly independent manner defined in this paper, then

$$(H_k - I)\frac{s_k}{|s_k|} = PP^t(H_k - I)Pr_k + O(\xi_k),$$

where $\xi_k \to 0$. Therefore by the results of this paper the convergence is superlinear if and only if $P^t H_k P \to I_M$.

Another motivation of this research was the desire to understand the question of superlinear convergence for sequential quadratic programming (SQP) methods for constrained optimization. The necessary and sufficient condition analogous to (8.1) for superlinear convergence is

$$\lim_{k\to\infty} PP^t(H_k - L)\frac{s_k}{|s_k|} = 0, \tag{8.2}$$

where L is the Hessian of the Lagrangian function with respect to the decision variable at the solution and PP^t is the projection onto the null space S of the active gradients at the solution. (See [11] or [12].) Note that the restriction of L to the subspace S is positive definite by the second-order sufficient conditions. Thus, letting

$$s_k = PP^t s_k + QQ^t s_k = r_k + v_k,$$

(8.2) becomes

$$\lim_{k\to\infty} \left\{ (PP^t(H_k - L)PP^t)\frac{r_k}{|s_k|} + (PP^t(H_k - L)QQ^t)\frac{v_k}{|s_k|} \right\} = 0. \tag{8.3}$$

It can be seen that if the sequence $\{s_k\}$ spans $\mathcal{R}^N$ in a K-step uniformly linearly independent manner, then the matrices $P^t(H_k - L)P$ and $P^t(H_k - L)Q$ must converge to zero if (8.3) is to hold. But L need not be positive definite except on the subspace S. Thus a BFGS or DFP updating scheme initiated with a positive definite matrix cannot be expected to lead to a superlinearly convergent algorithm unless the problem is convex (of course, it may not be possible to maintain a positive definite matrix in the nonconvex case). On the other hand, it is seen that if the sequence $\{r_k\}$ spans S in a uniformly linearly independent manner and $\{v_k\}$ tends to zero R-linearly, then the results of this paper imply that $PP^t(H_k - L)PP^t \to 0$ and hence superlinear convergence occurs. The sequence of vectors $\{v_k\}$ tending to zero is implied by the sequence of iterates $\{x_k\}$ tending to the solution in a "tangential" sense relative to S and so this result reinforces and explains previous observations (see [13]) that tangential convergence of the iterates in an SQP algorithm implies superlinear convergence. The remarks in §7 concerning the PSB update suggest that superlinear convergence can be obtained by an SQP algorithm employing that update, since under the assumption of uniform linear independence of the sequence $\{s_k\}$, the PSB updates converge to the Hessian matrix without the assumption of positive definiteness. Indeed, Han in his early work on the SQP method [14] was able to prove superlinear convergence for this update.

It would be ideal to be able to use the convergence analysis of this paper to deduce the "best" update scheme for a given optimization algorithm. This is unreasonable, however, since there are many other factors that influence a numerical algorithm that are not taken into account here. More significantly, our analysis treats the sequence of steps $\{s_k\}$ as generated independently of the sequence $\{H_k\}$; whereas in optimization algorithms the determination of s_{k+1} is directly influenced by H_k. This being said, it is hoped that some insight into the appropriateness or limitations of a particular update in a certain situation may be gained from this type of analysis. For example, the remarks above suggest that the updating methods that preserve positive definiteness, long favored for unconstrained optimization algorithms, cannot guarantee superlinear convergence when applied in SQP algorithms for constrained optimization problems.

One intriguing question that has inspired many research efforts (e.g., [6]) is to explain the experimentally observed superiority of the BFGS updating scheme over the DFP scheme in quasi-Newton algorithms for unconstrained optimization problems. In the context of system (2.2), the BFGS update corresponds to the C update and the DFP to the L update (where H_k

represents a direct approximation of the Hessian of the objective function). If one interprets the system (2.2) as a local approximation to the system (1.5), then one might hope to observe better convergence results for the C update than for the L update. However, according to Theorem 7.4, their theoretical convergence properties are identical for positive definite initial matrices in our format.

Another way to compare the local convergence properties of the Broyden class of updates is in terms of their relation to the SR1 update. As was noted in §2 the SR1 updating scheme used in (2.2) yields finite convergence in N steps (if $Es \neq 0$ at an earlier step) if the sequence $\{s_k\}_{k=0}^{N-1}$ is linearly independent, while the other updates require that the s_k be orthogonal (conjugate with respect to F) to obtain the same result. This property can be expressed in terms of the system (2.5) by observing that if s_{k-1}, s_k, and E_k are given with $E_k s_{k-1} = 0$, $s_k = \alpha s_{k-1} + v$, and $v^t s_{k-1} = 0$ then, in the notation of (6.1),

(8.4)

$$\begin{aligned} E_{k+1} &= G(s_k; E_k) - \phi_k H(s_k; E_k) = G(s_k; E_k) - \phi_S H(s_k; E_k) + (\phi_S - \phi_k) H(s_k; E_k) \\ &= G(v; E_k) - \phi_S H(v; E_k) + (\phi_S - \phi_k) H(s_k; E_k) \end{aligned}$$

where ϕ_S is the SR1 update parameter. Since the last term is zero if $\phi_k = \phi_S$, it is seen that the SR1 parameter is the only one that causes the update formula to be a function of only the component, v, of s_k perpendicular to the preceding step, s_{k-1}. Thus we can think of the last term involving $(\phi_S - \phi_k)$ as the error in the update that is preventing finite convergence. Clearly, in terms of speeding up convergence for system (2.5) (and hence (2.2)) it is best to choose $\phi_k = \phi_S$. If, however, it is desired that positive definiteness of the H_k matrices be preserved, then the value ϕ_S cannot always be chosen. A reasonable strategy in this case might be to choose the value of ϕ_k as close to ϕ_S as possible, consistent with preserving positive definiteness, thus minimizing the error term in (8.4). From Proposition 2.1 it is seen that positive definiteness is preserved if and only if $\phi \leq \phi_C$ where ϕ_C is a constant greater than one whose exact value depends on the current s and H. For the system (2.2)

$$\phi_S = \frac{s^t H s}{s^t H s - 1}$$

so that if $s^t H s < 1$, then $\phi_S < 0$ and hence positive definiteness is preserved, while if $s^t H s > 1$, then $\phi_S > 1$ and positive definiteness may or may not be preserved depending on the value of ϕ_C. These observations lead to the consideration of four strategies for choosing the update parameter to maintain positive definiteness for the system (2.2):

Strategy I: (pure DFP) Choose $\phi = 0$;

Strategy II: (pure BFGS) Choose $\phi = 1$;

Strategy III: (mixed BFGS and DFP) If $s^t H s < 1$, choose $\phi = 0$; otherwise choose $\phi = 1$;

Strategy IV: (mixed BFGS and SR1) If $s^t H s < 1$, choose $\phi = \phi_S$; otherwise choose $\phi = 1$.

Based on the arguments given above, one would expect Strategy IV on average to lead to faster convergence of the matrices than Strategy III and the latter to be better than either of the strategies I or II. Very limited numerical experimentation on randomly generated problems has supported these conjectures. This testing also gave an edge to Strategy II over Strategy I. A reasonable explanation for this is that the eigenvalues of the randomly generated initial matrices were preponderantly greater than one and, hence, so were those of the succeeding matrices. Therefore, on average, $s^t H s$ was more often than not greater than one, and thus the value $\phi = 1$ was closer to ϕ_S more often than was the value $\phi = 0$. This situation would also be expected to occur for almost any realistic distribution of positive definite matrices; thus one

should expect that the strategies above are listed in increasing order of general effectiveness. In particular, the BFGS strategy should generally out perform the DFP strategy when there are numerous large eigenvalues (cf. [6]). Presumably, the best strategy would be to choose ϕ as close to the upper bound ϕ_C as possible whenever ϕ_S is greater than ϕ_C and $\phi = \phi_S$ otherwise. Whether or not this is computationally attractive is not clear.

It should be emphasized that the above remarks apply only to the system (2.2) and not to the system (1.5). Further experimentation is necessary to determine if these conjectures can be validated for the latter system and applied in a useful way to quasi-Newton algorithms for unconstrained optimization.

A generalization of the results of this paper that also might yield interesting insights would be the relaxation of the secant condition (1.2). It seems clear that the analysis carried out here could also work if another term were added to (2.5), say $\hat{U}(s_k, E_k)$, with $\hat{U}s_k$ converging to, but not equal to, zero. Such a class of updates satisfying a type of asymptotic secant condition would yield the same convergence results and, by the remarks above, preserve superlinear convergence. Such updates could potentially broaden the choices of updating schemes available for use in special classes of problems.

A final question that is unanswered by this analysis is if the global convergence properties enjoyed by the L and C updates when $s_k = y_k$ can be extended to other updates. In particular, it would be reasonable to expect that the restricted Broyden class of updates would have this property.

Appendix. Here we address the proofs of the theorems in §3. We will not prove the theorems stated there in their full generality. Rather, we will sketch the proof of Theorem 3.1 and prove one theorem from which the proof of Theorem 3.2 can be derived by generalizing the arguments.

For Theorem 3.1 we consider two uncoupled systems

$$z_{k+1} = A_k z_k + Z_k(t_k)$$

and

$$w_{k+1} = w_k + W_k(t_k)$$

where, because of the R-linear convergence of the $\{t_k\}$,

$$|t_k| \leq |t_0| \cdot \tau^k \tag{A.1}$$

for some τ, $0 < \tau < 1$, and some t_0. For the first equation it can be shown by induction, using the fact that $\|A_i\| \leq 1$ for all i,

$$\begin{aligned} |z_{k+r}| &\leq \left(\left\| \prod_{j=0}^{r-1} (A_{k+j}) \right\| \right) |z_k| + \kappa_1 \cdot \sum_{j=0}^{r-1} |t_{k+j}| \\ &\leq \beta_{k,r} |z_k| + \frac{\kappa_1 \cdot |t_0|}{1-\tau} \tau^k. \end{aligned} \tag{A.2}$$

Assuming, without loss of generality, that $k_{j+1} \geq k_j + m$, condition (ii) of Theorem (3.1) implies

$$|z_{k_{j+1}}| \leq \beta |z_{k_j}| + K \cdot \tau^j$$

where $K = \kappa_1 \cdot |t_0|/(1-\tau)$. Let $\{\xi_j\}$ satisfy

$$\xi_{j+1} = \beta \xi_j + K \cdot \tau^j \tag{A.3}$$

with $\xi_0 = |z_{k_0}|$. Then for each j, $|z_{k_j}| \le \xi_j$. The solution to (A.3) satisfies

$$\xi_j = \beta^j \xi_0 + K \cdot \sum_{i=1}^{j} \beta^{j-i} \tau^i \le \rho^j [\xi_0 + K \cdot j]$$

where $\rho = \max[\beta, \tau]$. So from (A.2), for $k_j < k \le k_{j+1}$,

$$|z_k| \le |z_{k_j}| + K \cdot \tau^{k_j} \le \rho^j [|z_{k_0}| + K \cdot (j+1)].$$

Thus $\{z_k\} \to 0$. If condition (ii)′ of Theorem 3.1 holds, then without loss of generality, we can assume that it holds for all k. Then for $k = j \cdot m + r$, $0 < r \le m$, we have from (A.2)

$$|z_k| = |z_{r+j\cdot m}| \le \beta |z_{r+(j-1)\cdot m}| + K \cdot \tau^{r+(j-1)\cdot m}.$$

It can now be established by induction on j that for each r, $0 < r \le m$,

$$|z_{r+j\cdot m}| \le K_r \rho^{r+j\cdot m}$$

where K_r is a positive constant depending only on r and $\rho = \max[\beta, \tau]$. The R-linear convergence follows. If $Z_k = 0$, then for each k

$$|z_{k+m}| \le \beta |z_k|,$$

which gives the m-step convergence.

For the second system, to show that $\{w_k\}$ converges to some vector R-linearly, we note that for any positive integers k and i,

$$|w_{k+1} - w_k| \le \kappa_2 \cdot |t_0| \cdot \tau^k \cdot \sum_{j=0}^{i-1} \tau^j \le K \tau^k,$$

and hence $\{w_k\}$ is a Cauchy sequence converging to some w^*. Since

$$|w^* - w_k| = \lim_{i \to \infty} |w_{k+i} - w_k| \le K \tau^k,$$

it follows that the convergence is R-linear.

To show how Theorem 3.2 can be derived, we consider the system of coupled difference equations in two variables:

$$\begin{aligned} z_{k+1} &= \alpha z_k + \beta_1 (z_k)^2 + \gamma_2 \cdot (z_k + w_k) \cdot t_k, \\ w_{k+1} &= w_k + \beta_2 (z_k)^2 + \gamma_2 \cdot (z_k + w_k) \cdot t_k, \end{aligned} \tag{A.4}$$

where $0 < \alpha < 1$, β_1, β_2, γ_1, and γ_2 are nonnegative constants, and $\{t_k\}$ converges to zero R-linearly, i.e., (A.1) holds.

THEOREM A.1. *There exists positive constants ξ_0 and τ_0 such that if $|z_0| < \xi_0$ and $|t_0| < \tau_0$, then $|z_k| \le |z_0|$ and $|w_k| \le 3 \cdot |w_0|$ for all k. Moreover, there exists a w^* such that*

$$\{z_k\} \to 0 \quad \textit{and} \quad \{w_k\} \to w^*$$

R-*linearly*.

Proof. First we show, by induction, that the hypotheses imply that $|w_k| \le 3 \cdot |w_0|$ and that there is a ρ, $0 < \rho < 1$, such that

$$|z_k| \le |z_0| \rho^k.$$

We let $\beta = \max\{\beta_1, \beta_2\}$ and $\gamma = \max\{\gamma_1, \gamma_2\}$ and set

(i) $\hat{\alpha} = (1+\alpha)/2$,

(ii) $\rho = \max\{\hat{\alpha}^{1/2}, \tau^{1/2}\}$,

(iii) $\xi_0 = \min\left\{\dfrac{1-\alpha}{2\beta}, \left[\dfrac{|w_0|\cdot(1-\rho^2)}{\beta}\right]^{1/2}, |w_0|\right\}$,

(iv) $\tau_0 = \min\left\{\dfrac{(1-\rho)|z_0|}{8|w_0|\gamma}, \dfrac{(1-\rho)^2}{4\gamma}\right\}$.

Clearly the inequalities are true for $k = 0$. Assume them to be true for $j = 0, 1, \ldots, k$. Then using (i), (iii), and the induction hypotheses

$$\begin{aligned} |z_{k+1}| &\le \alpha|z_k| + \beta|z_k|\cdot|z_k| + \gamma(|z_k| + |w_k|)|t_0|\tau^k \\ &\le \hat{\alpha}|z_0|\rho^k + 4\gamma|w_0|\cdot\tau_0\cdot\tau^k. \end{aligned}$$

Using (ii) and (iv), we have (since $k \ge 1$)

$$|z_{k+1}| \le \rho^{k+1}\left\{\rho|z_0| + \frac{\rho(1-\rho)|z_0|}{2}\right\} \le |z_0|\rho^{k+1}. \tag{A.5}$$

Then for each j, $j = 0, \ldots, k$, we have

$$\begin{aligned} |w_{j+1}| &\le |w_j| + \beta|z_0|^2\rho^{2j} + \gamma(|z_0|\rho^j + 3|w_0|)|t_0|\rho^{2j} \\ &\le |w_j| + \{|w_0|(1-\rho^2) + 4\gamma|w_0|\tau_0\}\rho^{2j} \\ &\le |w_j| + 2|w_0|(1-\rho^2)\rho^{2j}. \end{aligned} \tag{A.6}$$

Clearly, the sequence $\{|w_j|\}$ is majorized by the sequence $\{\zeta_j\}$ satisfying

$$\begin{aligned} \zeta_0 &= |w_0| \\ \zeta_{j+1} &= \zeta_j + 2\zeta_0(1-\rho^2)\rho^{2j} \end{aligned}$$

for $j = 0, \ldots, k$. But then

$$\zeta_{k+1} = \zeta_0 + \sum_{j=0}^{k}[\zeta_{j+1} - \zeta_j] = \zeta_0 + 2\zeta_0(1-\rho^2)\sum_{j=0}^{k}\rho^{2j} \le 3\zeta_0.$$

Thus

$$|w_{k+1}| \le 3\cdot|w_0|$$

which completes the induction step. Now it follows from (A.5) that $\{z_k\} \to 0$ at an R-linear rate. It remains to show that the $\{w_k\}$ converge to some vector R-linearly. From the difference equation for the $\{w_k\}$ and the estimates in (A.6) it is seen that for any positive integers k and i,

$$|w_{k+i} - w_k| \le 2|w_0|(1-\rho^2)\sum_{j=0}^{i-1}\rho^{2(k+j)} \le 2|w_0|\rho^{2k}$$

and hence $\{w_k\}$ is a Cauchy sequence converging to some w^*. Since

$$|w^* - w_k| = \lim_{i\to\infty}|w_{k+1} - w_k|,$$

it follows from the above inequality that the convergence is R-linear. □

Theorem 3.2 can now be proved by generalizing the above theorem to the case where z_k and w_k are vectors and the condition $0 < \alpha < 1$ is replaced by conditions (i) and (ii)′ of Theorem 3.1.

Acknowledgments. The results in §4 were enhanced by helpful comments from Richard Byrd and the referees.

REFERENCES

[1] J. Dennis and R. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, Englewood Cliffs, NJ, 1983.

[2] R. Fletcher, *Practical Methods of Optimization*, Vol. 1, *Unconstrained Optimization*, John Wiley, Chichester, England, 1980.

[3] G. Schuller, *On the order of convergence of certain quasi-Newton methods*, Numer. Math., 23 (1974), pp. 181–192.

[4] R. Ge and M. Powell, *The convergence of variable metric matrices in unconstrained optimization*, Math. Programming, 27 (1983), pp. 123–143.

[5] J. Stoer, *The convergence of matrices generated by rank-2 methods from the restricted β-class of Broyden*, Numer. Math., 44, (1984), pp. 37–52.

[6] R. Byrd, J. Nocedal, and Y. Yuan, *Global convergence of a class of quasi-Newton methods on convex problems*, SIAM J. Numer. Anal., 24 (1987), pp. 1171–1190.

[7] A. Conn, N. Gould, and Ph. Toint, *Convergence of quasi-Newton matrices generated by the symmetric rank one update*, Math. Programming, 50 (1991), pp. 177–196.

[8] J. Wilkinson, *The Algebraic Eigenvalue Problem*, Oxford University Press, London, 1965.

[9] J. Ortega, and W. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, New York, 1970.

[10] H. Walker, *Implementation of the GMRES method using Householder transformations*, SIAM J. Sci. Statist. Comput., 9 (1988), pp. 152–163.

[11] P. Boggs, J. Tolle, and P. Wang, *On the local convergence of quasi-Newton methods for constrained optimization*, SIAM J. Control Optim., 20 (1982), pp. 161–171.

[12] J. Stoer and R. Tapia, *On the characterization of Q-superlinear convergence of quasi-Newton methods for constrained optimization*, Math. Comput., 49 (1987), pp. 581–584.

[13] P. Boggs and J. Tolle, *A strategy for global convergence in a sequential quadratic programming algorithm*, SIAM J. Numer. Anal., 26 (1989), pp. 1–23.

[14] S. Han, *Superlinearly convergent variable metric algorithms for general nonlinear programming problems*, Math. Programming, 11 (1976), pp. 263–382.

[15] R. Byrd, private communication.

SIAM J. OPTIMIZATION
Vol. 4, No. 2, pp. 288–296, May 1994
© 1994 Society for Industrial and Applied Mathematics
004

CAN PARALLEL BRANCH AND BOUND WITHOUT COMMUNICATION BE EFFECTIVE?*

PER S. LAURSEN†

Abstract. The usual approach to parallel Branch and Bound is to employ dynamic exchange of subproblems during the search in order to ensure a uniform workload on all processors. It is a common assumption that static distribution of subproblems is "inherently inefficient."

A parallel Branch and Bound algorithm was implemented, which does not communicate during the search. A small number of subproblems are initially generated and distributed among the processors which then perform ordinary serial Branch and Bound. Since the value of the current best solution is not distributed during the search it is necessary to be able to provide a good initial solution.

The algorithm has been applied to three combinatorial optimization problems: the quadratic assignment problem, the graph partitioning problem, and the weighted vertex cover problem. For all three problems, an average processor utilization of 0.70–0.85 was observed. Parallel Branch and Bound without communication is thus *not* inherently inefficient.

Key words. parallel processing, combinatorial optimization, Branch and Bound

AMS subject classifications. 65Y05, 68Q22, 90C27

1. Introduction.

NP-complete combinatorial optimization problems cannot (at present) be solved in polynomial time. These problems must then be solved by use of some enumeration technique, typically Branch and Bound (see, e.g., Papadimitriou and Steiglitz [10] on Branch and Bound). Branch and Bound does, however, in most cases only enable us to solve moderately sized problem instances, due to the exponential increase in time consumption when problem size increases. Hence, there is increasingly more interest in parallel algorithms for Branch and Bound in order to utilize the vast potential computational power in distributed systems.

Branch and Bound per se is in fact fairly easy to parallelize, and impressive results have already been obtained by several researchers (see, e.g., Lüling and Monien [9], Clausen and Träff [3]). These results stem from algorithms that employ dynamic distribution of subproblems during the search. It has so far more or less tacitly been assumed that one *must* exchange subproblems and other data between the processors during the search, in order to obtain high efficiency. The drawback of the dynamic distribution approach is that one has to implement management of communication, which is a nontrivial task in a distributed system. Furthermore, one has to design a communication protocol to provide the processors with rules for sending and requesting subproblems, to exchange other data and for termination detection. It may also require heavy experimentation to "tune" the algorithm to optimal performance, since rules for subproblem exchange, etc. often depend on several parameters. But, as indicated before, you can then end up with a highly performing algorithm, typically exhibiting efficiencies on 0.95 and above. The aim of this paper is not to try to develop yet another complicated algorithm that may tie or improve such results, but to investigate whether a reasonably effective algorithm can be obtained by much simpler means. Obviously, a great simplification can be achieved by abandoning communication during the search.

2. Parallel Branch and Bound without communication.

The idea is indeed very simple; just create an initial pool of disjunct subproblems, distribute a number of subproblems to each processor, let each processor perform serial Branch and Bound on the pool assigned

* Received by the editors October 8, 1992; accepted for publication (in revised form) March 26, 1993. This research was financed by a grant from the Danish Natural Science Research Council.

† Department of Computer Science, University of Copenhagen, Universitetsparken 1, 2100 Copenhagen Ø, Denmark (svalle@diku.dk).

to it, and collect the solutions. Note that not even the value of the current best solution is exchanged during the search. Such an approach has of course been considered before, but has been rejected as "inherently inefficient." Still, no results have been presented to confirm this conjecture. Evidently, there are some problems connected with this approach.

In this simple algorithm, the value of the global current best solution is not distributed among the processors. This may cause the total amount of work to increase. However, if we start the algorithm by finding a very good (near-optimal) initial solution, the total amount of work will not increase much, since only few noncritical subproblems will be investigated.

More serious is the fact that we cannot in advance estimate the difficulty of (i.e., the time needed to process) a given subproblem. A quantitative measure of the difficulty is not easily found, so we cannot construct subproblem pools of equal difficulty. This will cause starvation during the search, since some processors will terminate prematurely. The only way to remedy this situation seems to be to exchange subproblems dynamically. However, we describe below two approaches for solving the starvation problem—at least to some extent—without subproblem exchange.

2.1. Avoiding starvation, first approach. If we wish to avoid starvation, we must be able to create initial subproblem pools of approximately equal difficulty. This in turn implies that we must find a quantitative measure for the expected time consumption for a subproblem. Such a measure, which we denote as a *work function*, must be a function of a number of attributes of the given subproblem. One attribute is always readily available; the bound function value. The size of the subspace spanned by a subproblem, i.e., the number of feasible solutions resulting from instantiation of the free variables in the subproblem, is often a simple function of the number of free variables and may thus also be available. Still, there are a number of problems for which the subspace size is not easily found (e.g., the weighted vertex cover problem) so we can in fact only rely on the bound function value. The work function ought to be a simple function of the bound function value. It is, however, not clear which function to choose. The only reasonable restriction on the choice is that a low bound function value should yield a high work function value and vice versa, since we are assuming a minimization problem. Denoting the bound function value for some subproblem s by b_s, we could choose, e.g., $w(s) = 1/b_s, w(s) = 1/\log(b_s), w(s) = 1/b_s^n$ or $w(s) = \exp(-kb_s)$. In [7], a number of experiments were performed on the quadratic assignment problem (QAP) using an exponential work function:

$$w(s) = e^{-k\frac{b_s}{fdn^2}}, \tag{2.1}$$

where f and d are the average values of the coefficients of the F and D matrices defining the QAP. Seemingly, it was in fact possible to use (2.1) as a reasonable quantitative measure of the difficulty. The graph for processor utilization averaged over ten QAPs (Fig. 1) behaves nicely; a distinct "plateau" of optimal k-values and clear deterioration for large values. The three graphs for three individual problems do, however, reveal that the behavior is somewhat more random for individual problems. If (2.1) was a consistent measure, that ought not to happen. Furthermore, we have tried to use $w(s) = 1/\log(b_s)$ as the work function. This work function has the effect of "squeezing" the bound function values into a very narrow interval—the difference between w-values being only a few percent—and is thus not a quantitative measure as such. This approach gave an average processor utilization of 0.88, which is higher than the utilization obtained for the optimal k-values in (2.1). Equation (2.1) can thus not be considered a good measure for difficulty. Similar experiments with the graph partitioning problem and the weighted vertex cover problem have confirmed that work functions similar to (2.1) are not generally valid. We can conclude that it is at least not trivial to find a consistent measure of difficulty.

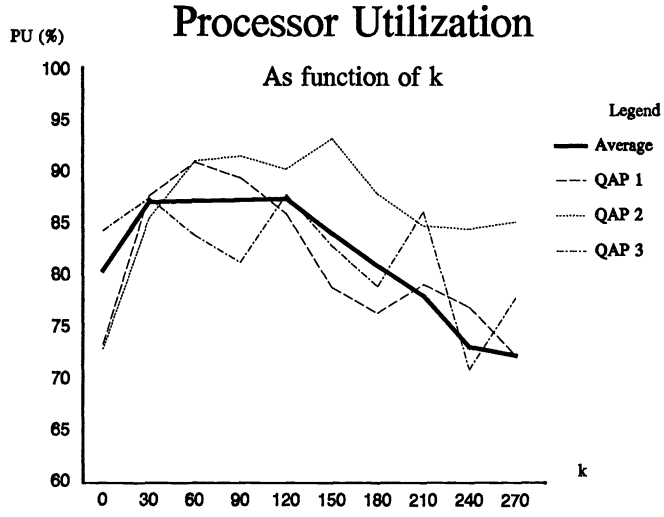


FIG. 1. *Performance for* QAP *using a work function.*

2.2. Avoiding starvation, second approach. Thus it seems that the bound function value cannot be taken to provide more than a "hint" of the difficulty of a subproblem. To be more specific; we can assume that the following proposition is usually true:

$$b(s_1) \leq b(s_2) \Leftrightarrow w(s_1) \geq w(s_2), \tag{2.2}$$

i.e., that the work function is monotone and nonincreasing, but not anything stronger than that. The goal must then be to create initial subproblems for which the bound function values cover as narrow an interval as possible. We can then conjecture that these subproblems will not differ substantially in difficulty. This objective can be achieved by using the so-called best-first search (BeFS) strategy in the initial phase. In BeFS, the live subproblems are stored as a heap, on top of which the subproblem holding the lowest bound function is located. That subproblem is then subdivided, and the new subproblems are subsequently inserted into the heap. When the heap is large enough, we have an initial subproblem pool consisting of subproblems with approximately equal bound function value. Experiments will show if these subproblems are also of approximately equal difficulty. This approach has the advantage of being very general; there is no need for some special problem-specific branching strategy, and we are relieved of the task of finding quantitative measures for difficulty.

3. The algorithm. We are now able to write up the algorithm for parallel Branch and Bound without communication during the search. The algorithm consists of six steps as described below.

(1) Find a good initial solution. It is as mentioned above important to find a near-optimal initial solution, by use of an appropriate heuristic.

(2) Subdivide the original problem by use of BeFS, until a sufficient amount of subproblems have been obtained, e.g., an order of magnitude higher than the number of processors, p. Here we have created about twenty times as many subproblems as processors in the network.

(3) Create p subproblem pools. Even though we use BeFS, there will be differences in the bound function values. Therefore, the hardest subproblems—i.e., those with lowest bound function value—are paired with the easiest ones, in order to even out the differences in difficulty.

(4) Distribute a pool to each processor, along with the initial solution.

(5) Perform serial Branch and Bound on each processor, on the subproblem pool assigned to the processor.

(6) Collect the solution proposals from each processor, and choose the best solution proposal as the final (and of course optimal) solution.

Note the simplicity of the algorithm. If a serial Branch and Bound algorithm for the problem at hand is available, one needs only to implement distribution of pools and collection of solutions, which can be performed without use of elements from "real" parallel programming, such as buffering of messages, broadcasting, etc.

4. Experiments and results. We have implemented the algorithm given above on a network consisting of 17 T800 Transputers™. One of these is designated as the root processor, and performs the initial pool construction, distribution of pools, etc. The remaining 16 processors perform Branch and Bound. The general algorithm has been specialized to three diverse NP-complete optimization problems; the quadratic assignment problem (QAP), the graph partitioning problem (GPP), and the weighted vertex cover problem (WVCP).

4.1. The test problems. The QAP is defined by two real matrices F and D, both consisting of nonnegative elements and zero diagonal elements. A solution is represented by a permutation of the integers $N = \{1, 2, \ldots, n\}$. The objective is to find that permutation π which minimizes

$$c(\pi) = \sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij} f_{\pi(i)\pi(j)} \tag{4.1}$$

over all permutations Π of N. More information on the QAP is found in Burkard, Finke, and Rendl [2].

The GPP is defined by a graph $G = (V, E)$. The matrix C is defined as having $c_{ij} = 1$ if there is an edge connecting vertices i and j, and zero otherwise. The problem is to divide the vertices into two partitions of equal size, so that there are as few edges as possible between the partitions. A solution is thus represented by a 0-1 vector of length $|V|$. The objective is to find that vector b which minimizes

$$c(b) = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} |b_i - b_j| \tag{4.2}$$

under the constraint that the sum of the elements in b must be $|V|/2$, over all vectors $\{0, 1\}^{|V|}$. More information on the GPP is found in Garey and Johnson [4].

The WVCP is defined by a graph $G = (V, E)$ and a vector W of weights on the vertices V. A cover C is a subset of V such that all edges in E have at least one endpoint in C. The optimal cover is the cover with the lowest total weight. A solution is represented by a 0-1 vector of length $|V|$, indicating whether or not a vertex is included in C. The objective is to find that vector b which minimizes

$$c(b) = \sum_{i=1}^{n} w_i b_i \tag{4.3}$$

under the constraint that each edge must have at least one endpoint in C, over all vectors $\{0, 1\}^{|V|}$. More information on the WVCP is found in Berge [1].

In order to implement a Branch and Bound algorithm for each of these problems, one must employ a bound function. The bound functions used for these three problems are described in Gilmore [5], Clausen and Träff [3], and Lüling and Monien [9] for the QAP, GPP, and WVCP, respectively.

4.2. Details of the experiment. In order to examine the performance of the algorithm, we use two measures; the processor utilization and the efficiency.

For each processor, the processor utilization is the time spent on actively performing Branch and Bound, divided by the total time spent by the network to perform Branch and Bound, i.e., from the moment the pools are distributed to the moment all solution proposals have been gathered. The processor utilization for the network is then the average utilization measured for all 16 processors. The processor utilization displays directly how well we have succeeded in dividing the work between the processors. The processor utilization will always be in the interval [0;1], 1 indicating an optimal—i.e., full—processor utilization.

The most common performance measure for parallel algorithms is the speedup or the efficiency, the latter just being the speedup divided by the number of processors. If the total amount of work remains unchanged by parallelization, the efficiency equals the processor utilization. The usual way of measuring speedup is to compare the running time of the parallel algorithm with that of the best-known—i.e., fastest—serial algorithm. The best Branch and Bound algorithm is the Best-First version, in which only critical subproblems (subproblems that have bound function value equal to or below the cost of the optimal solution) are investigated. For the parallel Branch and Bound algorithm described here, that will only happen if the initial solution is optimal. The efficiency will thus be equal to or lower than the processor utilization. The usual strategy for measuring speedup is thus to solve the test problems by a serial Best-First based Branch and Bound algorithm, and then by the parallel algorithm. This approach has the drawback of limiting the investigations to problem instances that can be solved by a serial algorithm in reasonable time. To circumvent this problem, we have measured speedup in the following way; first, the new parallel algorithm was applied to the test problems. This provided us with the optimal solution to the test problems. We then applied the parallel algorithm again, but provided the algorithm with the optimal solution as the initial solution. Hence only the critical subproblems were now examined, i.e., exactly those subproblems that would be examined by the serial Best-First algorithm. In our simple parallel version of the algorithm, a processor can only be in two states, active or terminated, so a very precise estimate of the running time for the serial algorithm is then found simply by adding up the active time for all 16 processors. In fact, the running time for the Best-First serial algorithm would probably be a little longer than suggested by the estimate, due to heap administration. We can thus consider the estimate to be a very tight lower bound on the running time for the serial algorithm.

As indicated by Tables 1 and 2 below, different problem sizes have been investigated for each problem. For the graph problems (GPP and WVCP) we have also investigated the behavior with respect to the average density of the graphs. Ten problem instances have been randomly generated for each problem size, except QAPs of size 15 and GPPs of size 66; here only five instances were tested due to time considerations. Hence a total of 150 different problem instances have been tested.

As described in §3, the parallel algorithm can be divided into three main phases; finding an initial solution, creation of subproblem pools, and distributed Branch and Bound without communication. As described above, we define processor utilization (*PU*) as the utilization during the dedicated Branch and Bound phase. The average processor utilization for each problem type is given in Table 1 below. To illustrate how the total time is spent, we also provide the amount of time spent in each of the three phases (T_{init}, T_{pools}, and T_{BB}, respectively). Since we are interested in a very good initial solution, we spend a relatively large amount of time on obtaining an initial solution. For the QAP and the WVCP, we use simulated annealing [6], while we employ the Lin–Kernighan heuristic [8] for the GPP. We have been able to obtain initial solutions of very high quality; on the average, the cost of the initial solution was about 0.2% above the cost of the optimal solution. All times are measured in seconds.

TABLE 1
Performance during Branch and Bound.

	n	d	PU	subprobs.	T_{init}	T_{pools}	T_{BB}
QAP							
	12		0.749	17356	25,9	22,2	35,4
	14		0.726	354204	29,2	42,0	1169,5
	15		0.711	1438004	31,0	54,2	6177,8
GPP							
	40	5	0.787	20072	2,5	11,9	19,6
	40	7	0.809	106833	2,5	11,7	104,7
	50	3	0.721	8560	4,6	19,6	14,0
	50	5	0.740	201921	4,5	15,9	337,2
	60	3	0.744	33477	8,8	22,3	80,2
	60	5	0.757	1026172	8,5	20,3	2259,1
	66	5	0.759	2916405	12,0	23,8	8042,6
WVCP							
	100	30	0.830	7865	13,0	24,7	10,1
	120	10	0.740	302547	13,5	65,3	1437,2
	120	30	0.845	27286	14,4	35,1	56,0
	120	50	0.845	6739	15,1	29,4	8,8
	140	30	0.823	129773	15,7	50,1	386,8
	160	30	0.814	752163	17,0	68,5	2863,6

With respect to processor utilization, the results are fairly consistent. The processor utilization is usually in the interval [0.70;0.85], which is not quite as high as for dynamic exchange algorithms, but still impressive for such a simple and general approach. All of the 150 test runs exhibited utilizations in the interval [0.58;0.91], i.e., the algorithm seems to be reasonably robust.

It might seem odd that the results for the QAP are poorer than those presented in connection with the work function. The explanation is that the previous approach was very problem-specific, which due to the special branching scheme employed for the QAP has some impact on the results. The branching scheme generates up to n new subproblems from a given subproblem, which makes it possible to design a superior, but problem-specific, initial subdivision. We must in some cases pay a price to obtain a more general approach.

The processor utilizations indicate that the initial phase of the algorithm is indeed capable of generating subproblem pools of approximately equal difficulty. The time used to perform this initial division constitutes for the smaller problems a significant part of the total time. For some problems, the time spent on the initial phases is actually longer than the time spent on Branch and Bound. Hence the total gain is not as high as indicated by the utilization. Table 2 below illustrates this by giving the (estimated) speedup and efficiency measures, both for the complete algorithm and for the Branch and Bound phase. The estimated serial time is denoted T_{ser}. Speedup and efficiency for the complete algorithm is denoted SU_{total} and E_{total}, respectively, while the same measures for the Branch and Bound phase are denoted SU_{BB} and E_{BB}, respectively.

These results stress that the algorithm is—in its present form—best suited for solving relatively hard problems. For the harder problems (more than 100,000 subproblems examined; see Table 1), the total efficiency is always above 0.60, which in our opinion is acceptable. For the easier problems, the efficiency is considerably lower. We could probably improve the performance for the easier problems by reducing the time spent on the initial phase, but that is of course an optimization problem in itself.

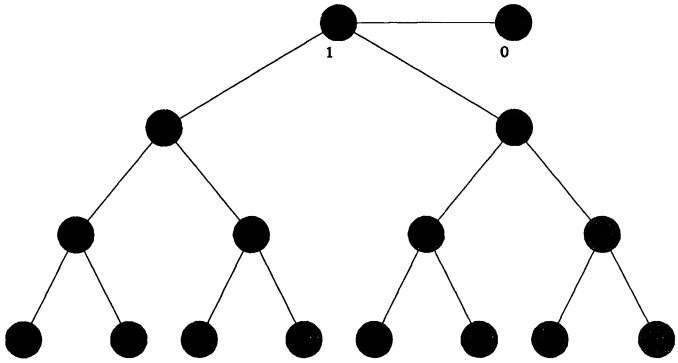


FIG. 2. *Binary processor tree with extra root.*

TABLE 2
Speedup and efficiency.

	n	d	T_{ser}	SU_{total}	E_{total}	SU_{BB}	E_{BB}
QAP							
	12		432,1	5.17	0.323	12.21	0.763
	14		12254,6	9.88	0.617	10.48	0.655
	15		64684,9	10.33	0.646	10.47	0.654
GPP							
	40	5	249,7	7.34	0.459	12.74	0.796
	40	7	1385,6	11.65	0.728	13.23	0.827
	50	3	156,2	4.09	0.256	11.16	0.697
	50	5	3958,9	11.07	0.692	11.74	0.734
	60	3	896,3	8.05	0.503	11.18	0.698
	60	5	27282,3	11.92	0.745	12.07	0.755
	66	5	95495,9	11.82	0.739	11.87	0.742
WVCP							
	100	30	125,7	2.63	0.164	12.45	0.778
	120	10	15004,6	9.90	0.619	10.44	0.653
	120	30	721,0	6.83	0.427	12.88	0.805
	120	50	108,3	2.03	0.127	12.31	0.769
	140	30	4665,7	10.31	0.644	12.06	0.754
	160	30	31923,6	10.82	0.677	11.15	0.697

5. Massive parallelization. We face another problem if we wish to use the algorithm for a larger number of processors. If we increase the number of processors, we obviously need to create more subproblems during the initial phase. The time spent on the initial phase will for a given problem increase like $O(p)$, while the total time will decrease like $O(1/p)$. This is clearly not acceptable, so the initial phase must be redesigned for large networks. Parallelizing the initial phase itself seems to be the obvious answer. A simple parallel algorithm for the initial phase is described below, but it sets two demands for the network topology.

(1) The number of processors p must be a power of two.

(2) A binary tree with an extra root must be embedded in the network topology, as illustrated in Fig. 2 above.

These two demands are not conceived to be severe, since future parallel systems will probably have the capability of being "virtually configurable" into any desired configuration. The algorithm then proceeds as follows.

First, the original subproblem is placed on processor 1, and subdivided into $2m$ subproblems (where m is, e.g., 20) by use of BeFS. The pool is then split into two pools, which are distributed to the two children of the processor. Each child will thus receive m subproblems, which are subsequently subdivided into $2m$ subproblems. These subproblems are now forwarded as before, and the process continues until m subproblems reach each of the "leaf processors," e.g., processors without children.

Second, each leaf processor subdivides its subproblems until it has $2m$ subproblems. Due to the BeFS search strategy, all of the pm subproblems now alive in the network ought to have approximately the same bound function value. Each leaf processor now forwards m of its subproblems to the parent processor, and keeps the remaining m subproblems for itself. Since each nonleaf processor has two children, each processor will receive $2m$ subproblems. They forward m subproblems as well, until $2m$ subproblems reach processor 1.

Third, processor 1 forwards m subproblems to the extra root processor, processor 0, implying that all processors in the network now possess m subproblems, which ought to be of comparable difficulty. The Branch and Bound part can thus be initiated.

The time needed to perform the initial phase in parallel is only $O(\log(p))$, which is clearly more acceptable than $O(p)$, especially when considering large ($p > 100$) networks.

6. Conclusion. We have examined the feasibility of excluding communication from parallel Branch and Bound in order to provide a simple, parallel algorithm. The algorithm was applied to three diverse optimization problems. The experiments revealed that parallel Branch and Bound without communication during the search is *not* inherently inefficient. The performance was not as high as for dynamic exchange algorithms, but the algorithm did show a consistent processor utilization on 0.70-0.85 for all three optimization problems. The efficiency indicated that the algorithm is best suited for solving hard problem instances.

It was, however, also stressed that the algorithm in its present form will decrease in performance if applied to large parallel systems. Parallelization of a now serial part of the algorithm was devised, which set certain hardware prerequisites. Experiments with the revised algorithm awaits availability of such parallel systems.

A straightforward enhancement of the present algorithm could be to include some simple distribution of the value of the global best solution, e.g., by circulation in an embedded ring. This would remove the necessity of a good initial solution, and could be implemented without demanding communication management as such.

The realm of applications, in the author's opinion, is in two areas; first, if one deals with a distributed system where communication is very expensive, or second—and perhaps most important—if one wants to utilize the potential computational power of a distributed system, but has no wish (or time) to employ the complicated machinery of parallel programming.

REFERENCES

[1] C. Berge, *The Theory of Graphs and Its Applications*, Methuen, London, 1962.

[2] R. E. Burkard, G. Finke, and F. Rendl, *Quadratic assignment problems*, Ann. Discrete Math., 31 (1987), pp. 61–82.

[3] J. Clausen and J. L. Träff, *Implementation of parallel branch and bound algorithms—experiences with the graph partitioning problem*, Ann. Oper. Res., 33 (1991), pp. 331–349.

[4] M. R. Garey and D. S. Johnson, *Computers and Intractability. A Guide to the Theory of* NP*-completeness*, Freeman, San Francisco, 1979.

[5] P. C. Gilmore, *Optimal and suboptimal algorithms for the quadratic assignment problem*, SIAM J. Appl. Math., 10 (1962), pp. 305–313.

[6] P. J. M. LAARHOVEN AND E. H. L. AARTS, *Simulated Annealing: Theory and Applications*, Reidel, Dordrecht, the Netherlands, 1987.
[7] P. S. LAURSEN, *Simple approaches to parallel Branch and Bound*, Parallel Computing, 19 (1993), pp. 143–152.
[8] S. LIN AND B. W. KERNIGHAN, *An efficient heuristic procedure for partitioning graphs*, Bell Systems Tech. J., 49 (1970), pp. 291–307.
[9] R. LÜLING AND B. MONIEN, *Load balancing for distributed Branch and Bound algorithms*, Proceedings of the 6th International Parallel Processing Symposium, 1992, pp. 543–549.
[10] C. H. PAPADRIMITRIOU AND K. STEIGLITZ, *Combinatorial Optimization, Algorithms and Complexity*, Prentice-Hall, Englewood Cliffs, NJ, 1982.

SIAM J. OPTIMIZATION
Vol. 4, No. 2, pp. 297–330, May 1994
© 1994 Society for Industrial and Applied Mathematics
005

CONVERGENCE THEORY OF NONLINEAR NEWTON–KRYLOV ALGORITHMS*

PETER N. BROWN† AND YOUCEF SAAD‡

Abstract. This paper presents some convergence theory for nonlinear Krylov subspace methods. The basic idea of these methods, which have been described by the authors in an earlier paper, is to use variants of Newton's iteration in conjunction with a Krylov subspace method for solving the Jacobian linear systems. These methods are variants of inexact Newton methods where the approximate Newton direction is taken from a subspace of small dimension. The main focus of this paper is to analyze these methods when they are combined with global strategies such as linesearch techniques and model trust region algorithms. Most of the convergence results are formulated for projection onto general subspaces rather than just Krylov subspaces.

Key words. nonlinear systems, nonlinear projection methods, Krylov subspace methods, inexact Newton methods, trust region techniques, conjugate gradient methods

AMS subject classification. 65H10

1. Introduction.

In a previous paper [5] we proposed several basic methods based upon the idea of employing a Newton iteration in which the Jacobian equations are solved approximately by a Krylov subspace method. Several theoretical issues raised in [5] were left unanswered. The purpose of this paper is to fill this gap by laying down the theoretical foundation of nonlinear Krylov subspace methods and by providing convergence results for them. In fact we will not limit ourselves to Krylov subspace methods. Rather, we discuss inexact Newton methods based on general projection techniques.

When defining algorithms for solving nonlinear systems of equations there are two possible options. First, one can use one of the globally convergent modifications of Newton's iteration [11]. The linear systems that arise in the course of the Newton iteration can be solved by either a direct solver or they may be solved approximately by an iterative method. The class of methods based on the latter approach is a particular case of *inexact Newton methods* and several such methods were considered in [1], [3], [4], and [5]. Newton's method is essentially a linearization procedure. The mapping F is locally approximated by a linear function and the resulting linear equations are solved to yield the next point. The second approach to solving nonlinear equations does not rely on linearization. Fixed-point iterations, for example, are inherently nonlinear as are descent methods with accurate line searches and the nonlinear conjugate gradient iteration.

We restrict our attention here to the first approach. In particular, we use linear Krylov methods to solve approximately the Newton equations. The Krylov methods considered are *Arnoldi's method* [27], and the *Generalized Minimum Residual Method* (GMRES) [28]. In general, these methods have the virtue of requiring virtually no

* Received by the editors December 21, 1989; accepted for publication (in revised form) January 5, 1993.

† Applied Mathematical Sciences Research Program. Computing and Mathematics Research Division, L-316, Lawrence Livermore National Laboratory, Livermore, California 94550. This author's work was performed under the auspices of the U.S. Department of Energy by the Lawrence Livermore National Laboratory under contract W-7405-Eng-48, and supported by the DOE Office of Energy Research.

‡ University of Minnesota, Computer Science Department, Minneapolis, Minnesota 55455. This author's work was supported by Cooperative Agreement NCC 2-387 between the National Aeronautics and Space Administration (NASA) and the Universities Space Research Association (USRA).

matrix storage, and therefore have a distinct advantage over direct methods for large scale problems.

To be more specific, consider finding a solution u_* of the nonlinear system of equations

$$F(u) = 0, \tag{1}$$

where F is a nonlinear function from $\mathbb{R}^N$ to $\mathbb{R}^N$. Newton's method applied to (1) results in the following iteration:

1. Set $u_0 =$ an initial guess.
2. For $n = 0, 1, 2, \ldots$ until convergence do the following:

$$\begin{aligned} &\text{Solve } J(u_n)\delta_n = -F(u_n), \\ &\text{Set } u_{n+1} = u_n + \delta_n, \end{aligned} \tag{2}$$

where $J(u_n) = F'(u_n)$ is the system Jacobian. For the problems under consideration, N is large, and as a result the so-called *inexact Newton* methods [9] solve (2) only approximately.

One of the main advantages of the Krylov methods considered here is that only the action of the Jacobian matrix J times a vector v is required, not J explicitly. In the current setting, this action can be well approximated by a difference quotient of the form

$$J(u)v \approx \frac{F(u + \sigma v) - F(u)}{\sigma}, \tag{3}$$

where u is an approximation to a solution of (1), and σ is a scalar. There are numerous applications where the Jacobian matrix is not practically computable, or is expensive to obtain. The most common of these are the so-called inverse problems, such as those arising in geophysics. Another typical example is in weather prediction as described in [22]. In addition, most of the nonlinear fixed-point iterations of the form $u_{n+1} = M(u_n)$ lead to such situations. In a nonlinear SSOR iteration for example, one single iteration is readily computable but it is not possible in general to determine the Jacobian of the mapping M associated with this iteration. To accelerate the fixed-point iteration one may solve the equation $u - M(u) = 0$ by a Krylov subspace method, yielding an inexact Newton method applied to $u - M(u) = 0$. Wigton, Yu, and Young [32], and Kerkhoven and Saad [17] have used this approach. Note that as was observed by Chan and Jackson [7], the new system of equations $u - M(u) = 0$ can be viewed as a nonlinearly preconditioned version of the original system of equations.

Here we will address the convergence behavior of the above algorithm when combined with a global linesearch backtracking strategy or model trust region approach. We should emphasize that our convergence results are not restricted to the use of Krylov subspace methods when solving (2). Our theory is formulated in terms of projection techniques wherein the approximation to the linear system (2) is taken from a small dimensional subspace.

Inexact Newton algorithms have been studied by several authors in recent years; see, for example, the references in [1] and the recent papers [6] and [13]. Several authors have considered using Krylov methods inside a Newton iteration in the context of systems of ordinary differential equations [3], [4], [7], [15]. Steihaug [30] and O'Leary [23] have used the *Conjugate Gradient* (CG) method in the unconstrained optimization of a real-valued function of several variables. Nash [20], [21] has also used a Newton–CG algorithm in unconstrained optimization.

In §2, we review inexact Newton algorithms, and present versions of the Newton–Arnoldi and Newton–GMRES methods. In §3 we give a convergence theory for inexact Newton methods when combined with a linesearch backtracking global strategy, and then in §4 we present a convergence theory for inexact Newton methods combined with model trust region strategies. In §5, we discuss applications of the basic results in the previous two sections to the Newton–Krylov methods, and then make some concluding remarks in §6.

2. Newton–Krylov methods. In this section we review some of the basic ideas of *inexact Newton methods* and *Newton–Krylov algorithms*. We begin with a discussion of the relevant results from Dembo, Eisenstat, and Steihaug [9] on inexact Newton methods, and then present the two inexact Newton methods we considered in [5], namely the Newton–Arnoldi and Newton–GMRES methods. Note that a Newton–Krylov method is one example of an inexact Newton algorithm.

2.1. Inexact Newton methods. From [9], an *inexact Newton method* for (1) has the following general form:

1. Choose u_0 an initial guess for u_*.
2. For $n = 0, 1, \ldots$ until convergence, do as follows:
   - Choose $\eta_n \in [0, 1)$.
   - Find (in some unspecified manner) a vector δ_n satisfying

$$J(u_n)\delta_n = -F(u_n) + r_n, \quad \text{with } \frac{\|r_n\|}{\|F(u_n)\|} \le \eta_n. \tag{4}$$

   - Set $u_{n+1} = u_n + \delta_n$.

The residual r_n represents the amount by which δ_n fails to satisfy the Newton equation (2). It is not generally known in advance, since it is the result of some inner algorithm which produces only an approximate solution to (2) (e.g., an iterative method). The forcing sequence $\eta_n \in (0, 1)$ is used to control the level of accuracy. Also, $\|\cdot\|$ represents any norm on $\mathbf{R}^N$.

We will make the following assumptions on F:

$$\begin{cases} \text{There exists a } u_* \in \mathbf{R}^N \text{ with } F(u_*) = 0. \\ F \text{ is } C^1 \text{ in a neighborhood of } u_*. \\ J(u_*) = F'(u_*) \text{ is nonsingular.} \end{cases} \tag{5}$$

The next theorem is shown in [9].

THEOREM 2.1. *Assume that F satisfies* (5)*, and that $\eta_n \le \eta_{\max} < t < 1$. There exists $\epsilon > 0$ such that if $\|u_0 - u_*\| \le \epsilon$, then the sequence of inexact Newton iterates $\{u_n\}$ converges to u_*. Moreover, the convergence is linear in the sense that*

$$\|u_{n+1} - u_*\|_* \le t\|u_n - u_*\|_*,$$

where $\|y\|_* \equiv \|J(u_*)y\|$. If in addition,

$$\eta_n \to 0, \tag{6}$$

then the sequence $\{u_n\}$ converges to u_* superlinearly. Also, if F' is Lipschitz continuous near u_* and $\eta_n = O(\|F(u_n)\|)$, then the convergence is quadratic.

In the above theorem, $\|\cdot\|$ again represents any norm on $\mathbf{R}^N$.

For the case when N is large, a δ_n satisfying the residual condition (4) is often obtained by using an iterative procedure for the linear system. In [5], we considered

using the Arnoldi and GMRES algorithms for nonsymmetric linear systems to obtain δ_n's satisfying the residual condition (4). For the convergence theory presented in this paper, the actual method which produces a δ_n satisfying (4) will be left unspecified. All that will be required is the existence of such a δ_n. This is easily guaranteed by assuming that J_n is nonsingular for all n.

2.2. Newton–Arnoldi and Newton–GMRES. At each iteration of the inexact Newton algorithm, we must obtain an approximate solution of the linear system (2) which we rewrite as

$$J\delta = -F, \tag{7}$$

where F and its Jacobian J are evaluated at the current iterate. If $\delta^{(0)}$ is an initial guess for the true solution of (7), then letting $\delta = \delta^{(0)} + z$, we have the equivalent system

$$Jz = r^{(0)}, \tag{8}$$

where $r^{(0)} = -F - J\delta^{(0)}$ is the initial residual. For a general N x N matrix A and vector v, define the *Krylov subspace* $K(A, v, m)$ by

$$K(A, v, m) = \text{span}\{v, Av, \ldots, A^{m-1}v\}.$$

Let K^m denote

$$K^m \equiv K(J, r^{(0)}, m).$$

Both Arnoldi's method and GMRES find an approximate solution

$$\delta^{(m)} = \delta^{(0)} + z^{(m)}, \quad \text{with } z^{(m)} \in K^m,$$

such that either

$$(-F - J\delta^{(m)}) \perp K^m \qquad (\text{equivalently } (r^{(0)} - Jz^{(m)}) \perp K^m) \tag{9}$$

for Arnoldi's method, or

$$\|F + J\delta^{(m)}\|_2 = \min_{\delta \in \delta^{(0)} + K^m} \|F + J\delta\|_2 \quad \left(= \min_{z \in K^m} \|r^{(0)} - Jz\|_2 \right) \tag{10}$$

for GMRES. Note that condition (10) is equivalent to demanding that the residual $r^{(m)} = -F - J\delta^{(m)}$ be orthogonal to JK^m. Here, $\|\cdot\|_2$ denotes the Euclidean norm on $\mathbf{R}^N$ and orthogonality is meant in the usual Euclidean sense.

The following algorithm is a nonlinear version of the Arnoldi (GMRES) algorithm, which at every outer iteration generates an orthonormal system of vectors v_i ($i = 1, 2, \ldots, m$) of the subspace K^m and then builds the vector $\delta^{(m)}$ that satisfies (9) (or (10) for GMRES). In both algorithms, v_1 is obtained by normalizing $r^{(0)}$.

ALGORITHM. Newton–Arnoldi (Newton–GMRES).

1. *Start:* Choose u_0 and compute $F(u_0)$. Set $n = 0$. Choose a tolerance ϵ_0.
2. *Arnoldi process:*
   - For an initial guess $\delta^{(0)}$, form $r^{(0)} := -F - J\delta^{(0)}$, where $F = F(u_n)$ and $J = J(u_n)$.
   - Compute $\beta := \|r^{(0)}\|_2$ and $v_1 := r^{(0)}/\beta$.

- For $j = 1, 2, \ldots,$ do:
  (a) Compute $w := Jv_j$
  (b) For $i = 1, \ldots, j$ do: $\begin{cases} h_{ij} := (w, v_i) \\ w := w - h_{ij} v_i \end{cases}$
  (c) Compute $h_{j+1,j} := \|w\|_2$ and $v_{j+1} = w/h_{j+1,j}$
  (d) Compute the residual norm $\rho_j = \|F + J\delta^{(j)}\|_2$, of the solution $\delta^{(j)}$ that would be obtained if we stopped at this step.
  (e) If $\rho_j \leq \epsilon_n$ set $m = j$ and go to (3).

3. *Form the approximate solution:*
   Arnoldi: Define H_m to be the m x m (Hessenberg) matrix whose (possibly) nonzero entries are the coefficients h_{ij}, $1 \leq i \leq \min\{j+1, m\}$, $1 \leq j \leq m$ and define $V_m \equiv [v_1, v_2, \ldots, v_m]$.
   - Findthe vector y_m which solves the linear system $H_m y = \beta e_1$, where $e_1 = [1, 0, \ldots, 0]^T$.
   - Compute $\delta^{(m)} := \delta^{(0)} + z^{(m)}$, where $z^{(m)} = V_m y_m$, and $u_{n+1} := u_n + \delta^{(m)}$.

   GMRES: Define $\bar{H}_m$ to be the $(m+1) \times m$ (Hessenberg) matrix whose (possibly) nonzero entries are the coefficients h_{ij}, $1 \leq i \leq j+1$, $1 \leq j \leq m$ and define $V_m \equiv [v_1, v_2, \ldots, v_m]$.
   - Find vector y_m which minimizes $\|\beta e_1 - \bar{H}_m y\|_2$, where $e_1 = [1, 0, \ldots, 0]^T$, over all vectors y in $\mathbf{R}^m$.
   - Compute $\delta^{(m)} = \delta^{(0)} + z^{(m)}$ where $z^{(m)} = V_m y_m$, and $u_{n+1} = u_n + \delta^{(m)}$.
4. *Stopping test:* If u_{n+1} is determined to be a good enough approximation to a root of (1), then stop, else set $u_n \leftarrow u_{n+1}$, $n \leftarrow n+1$, choose a new tolerance ϵ_n, and go to (2).

Therefore, in both Arnoldi and GMRES the outer iteration is of the form $u_{n+1} = u_n + \delta^{(m)}$ where $\delta^{(m)} = \delta^{(0)} + z^{(m)}$, with

$$z^{(m)} = V_m y_m,$$

and y_m is either the solution of an $m \times m$ linear system for Arnoldi, or the solution of an $(m+1) \times m$ least squares problem for GMRES.

For simplicity, we have omitted several details of the practical implementation of the above linear and nonlinear methods, which are discussed at length in [5], [27], [4], and [28]. For example, the residual norm ρ_j referred to in step 2 of the algorithm does not require the computation of the approximate solution $\delta^{(j)}$ at every step. Instead an inexpensive formula, which evaluates ρ_j, is updated at each step while the factorization of the Hessenberg matrix H_m or $\bar{H}_m$ is updated. Also, the choice of the sequence $\{\epsilon_n\}$ is typically of the form $\epsilon_n = \eta_n \|F(u_n)\|_2$, where $\eta_n \to 0$ as $n \to \infty$, although other choices are possible. We elaborate on possible choices for the sequence $\{\eta_n\}$ in §§3 and 4.

An important observation that has been very useful in practice is the implicit use of the Jacobian matrix $J(u_n)$ in the algorithms. This is due to the fact that only the product of $J(u_n)$ times a vector is required, and this can be well approximated by the difference formula (3). In [1], Brown has given an analysis of the resulting inexact Newton/finite-difference Krylov algorithms when using (3) to approximate $J(u)v$. Sufficient conditions are given in [1] on the size of the σ's in the finite-difference versions of Arnoldi and GMRES which guarantee the local convergence of the Newton–Krylov iteration. These results have been extended in [4] to include a finite-difference version of the Conjugate Gradient iteration.

One final aspect worth noting is the ability to use restarting in the linear Krylov methods. Typically, a maximum value of m is dictated by storage considerations. If we let $m_{\max}$ be this value, then it is possible that $m = m_{\max}$ in the Arnoldi process, and yet ρ_m is still greater than ϵ_n. In this case, one can set $\delta^{(0)}$ equal to $\delta^{(m)}$ and restart the Arnoldi process, effectively restarting the Krylov method. The convergence of such a procedure is not always guaranteed, but the idea seems to work well in practice. We note that for lack of a better initial guess we use $\delta^{(0)} = 0$ on the first (and possibly only) pass through the Arnoldi process at each stage of the Newton iteration. It is only when restarting that $\delta^{(0)}$ will be nonzero. We will refer to the restarted algorithms as Arnoldi(m) and GMRES(m), where m is the maximum subspace dimension. As will be seen below, it will also be important to choose the tolerance ϵ_n at each step of the Newton iteration.

3. Convergence results for linesearch methods. We will be concerned with the convergence properties of the inexact Newton algorithms outlined in the previous section when combined with global strategies. In this section we will analyze the global convergence of inexact Newton algorithms when combined with linesearch backtracking strategies. The results given below are valid for general inexact Newton methods.

To begin, let $f(u) = \frac{1}{2}\|F(u)\|_2^2$. An easy calculation gives

$$\nabla f(u) = J(u)^T F(u),$$

where $J(u) = F'(u)$, the Jacobian matrix of F evaluated at u. Typically, convergence of a sequence of iterates $\{u_n\}$ is studied in terms of the scalar sequence

$$\xi_n \equiv \nabla f_n^T \frac{\delta_n}{\|\delta_n\|_2}, \tag{11}$$

where

$$\delta_n \equiv u_{n+1} - u_n \text{ and } \nabla f_n = \nabla f(u_n).$$

When $\lim_{n\to\infty} \xi_n = 0$, the sequence u_n will converge to a solution u_* under fairly mild conditions. First, let us assume that the acute angle between δ_n and the gradient ∇f_n is bounded away from $\pi/2$, i.e., that at every step we have

$$\cos\theta(\nabla f_n, \delta_n) \geq \epsilon > 0, \tag{12}$$

(where we define $\cos\theta(u,v) = |u^T v|/\|u\|_2\|v\|_2$). Then from the definition of ξ_n in (11), the gradient ∇f_n will converge to zero whenever ξ_n converges to zero.

We now recall the following two important results from Ortega and Rheinboldt [24, pp. 475–476].

THEOREM 3.1. *Let $f : \mathbb{R}^N \to \mathbb{R}$ be continuously differentiable on a compact subset $D_0 \subset \mathbb{R}^N$ and suppose that $\{u_n\} \subset D_0$ is any sequence which satisfies $\lim_{n\to\infty} \nabla f(u_n) = 0$. Then the set $\Omega = \{u \in D_0 | \nabla f(u) = 0\}$ of critical points of f is not empty and*

$$\lim_{n\to\infty} [\inf_{u\in\Omega} \|u - u_n\|] = 0. \tag{13}$$

In particular, if Ω consists of a single point u_* then $\lim_{n\to\infty} u_n = u_*$ and $\nabla f(u_*) = 0$.

THEOREM 3.2. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable on a compact subset $D_0 \subset \mathbb{R}^N$ and suppose that the set $\Omega = \{u \in D_0 | \nabla f(u) = 0\}$ of critical points of f in D_0 is finite. Let $\{u_n\} \subset D_0$ be any sequence for which $\lim_{n\to\infty} \nabla f(u_n) = 0$ and $\lim_{n\to\infty}(u_{n+1} - u_n) = 0$. Then u_n converges to a certain u_* in Ω and $\nabla f(u_*) = 0$.*

Thus, we will often attempt to establish conditions under which ξ_n as defined by (11) converges to zero and for which (12) holds.

To guarantee that the current iterate will make progress towards the solution in one step of the algorithm we must know that the inexact Newton step δ is a descent direction for f at the current approximation u. A *descent direction* p at u is one for which there exists a $\lambda_0 > 0$ such that $f(u + \lambda p) < f(u)$ for all $\lambda < \lambda_0$. As is well known, when f is differentiable this is equivalent to the condition that

$$\nabla f(u)^T p < 0,$$

where $\nabla f(u) = ((\partial f/\partial u_1)(u), \ldots, (\partial f/\partial u_N)(u))^T$. As noted above, $\nabla f(u) = J(u)^T F(u)$, and so p is a descent direction for f at u if

$$F(u)^T J(u) p < 0.$$

If $\bar{\delta}$ is an approximate solution of the Newton equations

$$J\delta = -F,$$

with $F = F(u)$ and $J = J(u)$, then

$$F^T J\bar{\delta} = -F^T F - F^T \bar{r}, \tag{14}$$

where $\bar{r} = -F - J\bar{\delta}$ is the residual associated with $\bar{\delta}$. Thus $\bar{\delta}$ will be a descent direction for f at u whenever $|F^T \bar{r}| < F^T F$. In particular, if $\|\bar{r}\|_2 < \|F\|_2$, then $\bar{\delta}$ is a descent direction. This result was also given in [5] and is restated in the following proposition.

PROPOSITION 3.3. *A sufficient condition for $p \in \mathbb{R}^N$ to be a descent direction for f at u is that*

$$\|F(u) + J(u)p\|_2 < \|F(u)\|_2. \tag{15}$$

As was seen earlier, it is also important to be able to guarantee that the angle between the gradient of f and the step δ_n is bounded away from $\pi/2$. The next lemma gives a lower bound for $|\xi_n|$ under an additional assumption on the step direction p.

LEMMA 3.4. *Let $F : \mathbb{R}^N \to \mathbb{R}^N$ be continuously differentiable on $\mathbb{R}^N$. Let $u \in \mathbb{R}^N$ be given with $F(u) \neq 0$ and $J(u) = F'(u)$ nonsingular. Consider $p \in \mathbb{R}^N$ satisfying*

$$\|F(u) + J(u)p\|_2 \leq \eta \|F(u)\|_2, \tag{16}$$

with $\eta \in (0, 1)$. Then

$$\frac{|\nabla f(u)^T p|}{\|p\|_2} \geq \frac{1 - \eta}{(1 + \eta)M} \|\nabla f(u)\|_2 > 0, \tag{17}$$

where $M = \mathrm{cond}_2(J(u))$ and $f(u) \equiv \frac{1}{2} F(u)^T F(u)$.

Proof. For notational convenience, let $F = F(u)$, $f = f(u)$, $\nabla f = \nabla f(u)$, and $J = J(u) \equiv F'(u)$. Note that $F \neq 0$ implies $\nabla f \neq 0$. Let r be the residual associated with p so that $r = F + Jp$. Then $\|r\|_2 \leq \eta \|F\|_2$ and $p = -J^{-1}(F - r)$. So,

$$\begin{aligned} \nabla f^T p &= (J^T F)^T (-J^{-1}(F - r)) \\ &= -F^T F + F^T r. \end{aligned}$$

Hence,

$$\frac{|\nabla f^T p|}{\|p\|_2} = \frac{|F^T F - F^T r|}{\|J^{-1}(F-r)\|_2} \geq \frac{F^T F - |F^T r|}{\|J^{-1}(F-r)\|_2}.$$

Next, $\|r\|_2 \leq \eta \|F\|_2$ implies $|F^T r| \leq \eta \|F\|_2^2$, which then gives

$$F^T F - |F^T r| \geq (1-\eta)\|F\|_2^2.$$

Also,

$$\begin{aligned} \|J^{-1}(F-r)\|_2 &\leq \|J^{-1}\|_2 \cdot \|F\|_2 + \|J^{-1} r\|_2 \\ &\leq (1+\eta)\|J^{-1}\|_2 \cdot \|F\|_2. \end{aligned} \tag{18}$$

Thus,

$$\frac{|\nabla f^T p|}{\|p\|_2} \geq \frac{(1-\eta)F^T F}{(1+\eta)\|J^{-1}\|_2 \cdot \|F\|_2} = \frac{(1-\eta)\|F\|_2}{(1+\eta)\|J^{-1}\|_2} \tag{19}$$

and as a result, using the fact that $\|\nabla f\|_2 = \|J^T F\|_2 \leq \|J\|_2 \|F\|_2$, we get

$$\frac{|\nabla f^T p|}{\|\nabla f\|_2 \cdot \|p\|_2} \geq \frac{(1-\eta)}{(1+\eta)M}, \tag{20}$$

where $M = \mathrm{cond}_2(J)$. □

Condition (17) can also be recast as

$$\cos\theta(\nabla f, p) \geq \frac{1-\eta}{(1+\eta)M}. \tag{21}$$

At every step of the inexact Newton method, we require that a condition of the form

$$\|F(u_n) + J(u_n)p_n\|_2 \leq \eta_n \|F(u_n)\|_2, \tag{22}$$

$$\eta_n \leq \eta < 1 \tag{23}$$

holds, and if we assume that the condition numbers $M_n = \mathrm{cond}_2(J(u_n))$ are bounded from above by M, then (17) shows that

$$\cos\theta(p_n, \nabla f(u_n)) \geq \frac{1}{M}\frac{1-\eta}{1+\eta}. \tag{24}$$

This implies that a sufficient condition to guarantee both p_n being a descent direction and the validity of relation (12) is that the residual condition (22)–(23) holds.

A simple consequence of the above lemma which will be useful in the section on trust region techniques, is that the residual norm assumption (22)–(23) implies that the cosine of the angle between the gradient and the Krylov subspace is bounded from below. More specifically, we can state the following corollary.

COROLLARY 3.5. *Let $F : \mathbb{R}^N \to \mathbb{R}^N$ be continuously differentiable on $\mathbb{R}^N$. Let $u \in \mathbb{R}^N$ be given with $F(u) \neq 0$ and $J(u) = F'(u)$ nonsingular. Consider the subspace*

$K = \text{span}\{V\}$ where the columns of V form an orthonormal set of vectors, and assume that there exists one vector p in K satisfying

$$\|F(u) + J(u)p\|_2 \leq \eta \|F(u)\|_2, \tag{25}$$

with $\eta \in (0,1)$. Then

$$\|V^T \nabla f(u)\|_2 \geq \frac{1-\eta}{(1+\eta)M} \|\nabla f(u)\|_2, \tag{26}$$

where $M = \text{cond}_2(J(u))$ and $f(u) \equiv \frac{1}{2} F(u)^T F(u)$.

Proof. Let $p = Vy$ be a vector of K that satisfies (25). We then have from the lemma

$$\frac{|\nabla f(u)^T V y|}{\|y\|_2} \geq \frac{1-\eta}{(1+\eta)M} \|\nabla f(u)\|_2, \tag{27}$$

where we have used the fact that $\|p\|_2 = \|Vy\|_2 = \|y\|_2$. Using the Cauchy–Schwarz inequality,

$$\|V^T \nabla f(u)\|_2 \geq \frac{|\nabla f(u)^T V y|}{\|y\|_2} \geq \frac{1-\eta}{(1+\eta)M} \|\nabla f(u)\|_2, \tag{28}$$

which proves the result. □

3.1. Convergence of inexact Newton sequences. In this subsection we consider the case of linearly converging inexact Newton sequences. That is, the sequence $\{\eta_n\}$ in the inexact Newton method is only required to satisfy $\eta_n \leq \eta_{\max} < t < 1$. Superlinearly converging inexact Newton sequences will be examined in the following subsection.

An important condition to guarantee global convergence is the so-called α-condition in the Armijo–Goldstein principle [11], [24], wherein δ_n must satisfy

$$f(u_n + \delta_n) \leq f(u_n) + \alpha \nabla f(u_n)^T \delta_n. \tag{29}$$

We can show a remarkably simple result if we require that the direction δ_n solve the linear system $J(u_n)\delta = -F(u_n)$ with a certain accuracy. In inexact Newton methods, the direction δ_n is required to satisfy a *residual norm condition* of the form,

$$\|F(u_n) + J(u_n)\delta_n\|_2 \leq \eta \|F(u_n)\|_2, \tag{30}$$

in which $0 \leq \eta < 1$ is a certain tolerance used in some process that approximately solves the system (2). However, we start by observing that there are instances in which the above α-condition and the residual norm condition cannot be satisfied simultaneously.

LEMMA 3.6. *Let $f \equiv \frac{1}{2}\|F\|_2^2$, where F is differentiable, and a scalar η with $0 \leq \eta < 1$ be given. If $F(u_n) \neq 0$ and if the scalar α satisfies the relation,*

$$\alpha > \frac{1}{2(1-\eta)} \tag{31}$$

then there is no iterate u_{n+1} defined by $u_{n+1} = u_n + \delta_n$, in which δ_n satisfies both conditions (29) *and* (30).

Proof. In this proof we let $J_n \equiv J(u_n)$, $F_n \equiv F(u_n)$. Let us assume that there is a δ_n satisfying (30) and (29) at the same time. From the condition (29) and the expression for the gradient of f we get

$$f(u_{n+1}) \leq f(u_n) + \alpha \nabla f_n^T \delta_n = f(u_n) + \alpha F_n^T J_n \delta_n. \tag{32}$$

With $J_n\delta_n = -F_n + r_n$ this gives

$$\begin{aligned} f(u_{n+1}) &\leq f(u_n) + \alpha F_n^T(-F_n + r_n) \\ &= (1-2\alpha) f(u_n) + \alpha F_n^T r_n \\ &\leq (1-2\alpha) f(u_n) + \alpha \|F_n\|_2 \|r_n\|_2. \end{aligned}$$

From (30) we have $\|r_n\|_2 \leq \eta \|F_n\|_2$ which yields the following inequality,

$$f(u_{n+1}) \leq (1-2\alpha) f(u_n) + 2\alpha\eta f(u_n) = [1 - 2\alpha(1-\eta)] f(u_n). \tag{33}$$

If we now assume (31) holds then the scalar in the brackets is negative and if $F(u_n) \neq 0$ we will reach the contradiction that $f(u_{n+1}) < 0$. Therefore, there cannot be an iterate u_{n+1} that satisfies both (29) and (30) at the same time if (31) holds. □

We can now show the following result.

THEOREM 3.7. *Let $f \equiv \frac{1}{2}\|F\|_2^2$ be given, where F is differentiable, and α, η two scalars such that*

$$0 \leq \eta < 1 \quad \textit{and} \quad 0 < \alpha \leq \frac{1}{2(1-\eta)}\ . \tag{34}$$

Assume that the iterates u_n are defined by $u_{n+1} = u_n + \delta_n$ where δ_n satisfies (29) *and* (30). *Then*

$$\lim_{n \to \infty} f(u_n) = 0.$$

Proof. If (30) and (29) both hold then as was shown in the previous proof, inequality (33) holds. The rest of the proof is a simple consequence of (33) and the fact that under the assumption on α and η we have

$$0 \leq 1 - 2\alpha(1-\eta) < 1. \qquad \square$$

If, for a given η, the scalar α satisfies (31) then the above lemma indicates that we cannot construct the next iterate. An interesting additional observation is that when η is close to one then we are allowed by the statement of the theorem to have α larger than one. We will, however, always restrict α to be less than unity. Note that we have made virtually no assumption on the function F apart from differentiability, and so the result is very general. However, given a pair η, α satisfying (34) we cannot guarantee that one can indeed select a vector δ_n satisfying conditions (29) and (30) at the same time. To be more specific, when F is continuously differentiable and δ_n is a descent direction, one can show (see [11]) that there exists a sufficiently small λ_n for which

$$f(u_n + \lambda_n \delta_n) \leq f(u_n) + \alpha \nabla f(u_n)^T (\lambda_n \delta_n), \tag{35}$$

and for this λ_n it is clear that

$$\|F(u_n) + J(u_n)(\lambda_n \delta_n)\|_2 \leq [(1-\lambda_n) + \lambda_n \eta]\ \|F(u_n)\|_2. \tag{36}$$

As a result, the conditions (30) and (29) are simultaneously satisfied with δ_n replaced by $\lambda_n\delta_n$ and η replaced by $\eta_n = (1-\lambda_n) + \lambda_n\eta$. The difficulty is that this new η is now variable and may tend to one when $\lambda_n \to 0$, in which case the theorem does not apply. On the other hand, when $\alpha < \frac{1}{2}$, we do know that near a solution u_* for which $J(u_*)$ is nonsingular, a sufficiently good approximation to the Newton step will satisfy the two conditions of the theorem simultaneously. (See Theorem 3.15.)

A more explicit result extending the previous theorem is now shown. As was mentioned above the difficulty with Theorem 3.7 is that the standard condition (36) may lead to a sequence λ_n converging to zero, leading to a sequence η_n converging to 1. It may be possible to prevent this by using a general backtracking strategy. This means that the next iterate is of the form $u_n + \lambda_n p_n$, where p_n is any descent direction and λ_n is selected by the procedure described below. In the procedure the two parameters $\theta_{\min}, \theta_{\max}$ are such that $0 < \theta_{\min} \leq \theta_{\max} < 1$, a simple choice being $\theta_{\min} = \theta_{\max} = \frac{1}{2}$. Other more sophisticated choices fitting this framework are described in [11, §6.3.2.], The procedure requires another parameter $\epsilon^* > 0$ which is used to essentially rescale the *starting step* in the process in order to prevent it from from being too small. We emphasize here that in order to obtain the most general result we make no assumption in the algorithm below and the next lemma regarding the search direction p_n. In particular the residual norm condition (30) is not assumed.

ALGORITHM 3.1. General Backtracking Procedure.

1. Set $\lambda = \max\{1, \epsilon^* |\nabla f(u_n)^T p_n| / \|p_n\|_2^2\}$.
2. If $f(u_n + \lambda p_n) \leq f(u_n) + \alpha\lambda\nabla f(u_n)^T p_n$, then set $\lambda_n = \lambda$, and exit. Else:
3. Choose $\hat{\lambda} \in [\theta_{\min}\lambda, \theta_{\max}\lambda]$; set $\lambda \leftarrow \hat{\lambda}$. Go to (2).

As is shown next, the sequence is well defined in that under a mild condition on the gradient of f the procedure will deliver a nonzero λ_n in a finite number of steps. Moreover, the resulting λ_n can be bounded from below.

LEMMA 3.8. *Let f be differentiable and assume that its gradient is such that*

$$\|\nabla f(x) - \nabla f(y)\|_2 \leq \gamma \|x - y\|_2 \quad \text{for all } x, y \in \mathbb{R}^N. \tag{37}$$

Let $0 < \alpha < 1$ and p_n be any descent direction. Then Algorithm 3.1 will produce an iterate $u_{n+1} = u_n + \lambda_n p_n$ in a finite number of backtracking steps and λ_n satisfies the inequality

$$\lambda_n \|p_n\|_2 \geq -\frac{\nabla f(u_n)^T p_n}{\|p_n\|_2} \min\left\{\epsilon^* \, , \, \frac{(1-\alpha)}{\gamma}\theta_{\min}\right\}. \tag{38}$$

Proof. The subscript n is dropped from this proof. Using the mean value theorem we have the equality:

$$f(u + \lambda p) = f(u) + \lambda \nabla f(u + \theta\lambda p)^T p, \tag{39}$$

where $0 \leq \theta \leq 1$. We rewrite the above equation as

(40)

$$\begin{aligned} f(u + \lambda p) &= f(u) + \lambda \nabla f(u)^T p + \lambda[\nabla f(u + \theta\lambda p)^T p - \nabla f(u)^T p] \\ &= f(u) + \alpha\lambda \nabla f(u)^T p + \lambda[(1-\alpha)\nabla f(u)^T p + (\nabla f(u + \theta\lambda p)^T p - \nabla f(u)^T p)] \\ &= f(u) + \alpha\lambda \nabla f(u)^T p + \lambda[(1-\alpha)\nabla f(u)^T p + \lambda\|p\|_2 \zeta], \end{aligned}$$

where for convenience we have set

$$\zeta \equiv \frac{\nabla f(u+\theta\lambda p)^T p - \nabla f(u)^T p}{\lambda \|p\|_2}.$$

Note that from the assumptions we have

$$|\zeta| = |\left(\nabla f(u+\theta\lambda p) - \nabla f(u)\right)^T \frac{p}{\lambda\|p\|_2}| \leq \gamma\theta\|p\|_2 \leq \gamma\|p\|_2. \tag{41}$$

If the test in step 2 is passed at the first step, then the first λ is accepted and, in this situation, the inequality (38) is trivially satisfied. If the test in step 2 fails for the first step, then λ is reduced according to the rule in step 3. Moreover, after a finite number of reductions, the term in brackets in the right-hand side of (40) will become negative and the corresponding λ will be accepted. This will occur no later than when $\lambda\gamma\|p\|_2^2 < -(1-\alpha)\nabla f(u)^T p$. The first λ which is accepted will be such that

$$\lambda\|p\|_2 \geq -\theta_{\min}\frac{(1-\alpha)}{\gamma}\frac{\nabla f(u)^T p}{\|p\|_2},$$

and the inequality (38) is again satisfied. □

We note that the usual β condition of Armijo–Goldstein also guarantees that a lower bound on λ similar to (38) is satisfied. Indeed, the relation (40) is still valid with α replaced by β. Moreover, the β condition

$$f(u+\lambda p) \geq f(u) + \beta\lambda\nabla f(u)^T p$$

implies that

$$(1-\beta)\nabla f(u_n)^T p_n + \lambda_n\|p_n\|_2\zeta \geq 0.$$

With the inequality (41) this immediately yields

$$\lambda_n\|p_n\|_2 \geq -\frac{(1-\beta)\nabla f(u_n)^T p_n}{\gamma\|p_n\|_2}. \tag{42}$$

Both (38) and (42) imply that the step length from u_n is bounded from below with respect to $\nabla f(u_n)^T p_n/\|p_n\|_2$.

We emphasize the importance of the initial λ in the procedure. There is no reason why one should always start the process with $\lambda = 1$ since $\|p_n\|_2$ can be arbitrarily small. As was explained before, the choice of the initial λ in step 1 is essentially equivalent to a rescaling of the vector p_n. If we always start with $\lambda = 1$ and p_n happens to be very small at every step, then the test in step 2 may be passed immediately and there is a danger that δ_n becomes too small for the iterates to make any progress towards the solution. Note, however, that if p_n solves the linear system $Jp = -F$ approximately, then we may have additional information that will ensure that $\|p_n\|_2$ is bounded from below. Indeed, the following lemma shown by Walker [31] is just one such result.

LEMMA 3.9. *Suppose that* $J(u)p = -F(u) + r$ *and* $\|r\|_2 \leq \eta\|F(u)\|_2$, *with* $0 \leq \eta < 1$. *Then,*

$$\|p\|_2 \geq \frac{(1-\eta)}{\|J(u)\|_2^2}\frac{|\nabla f(u)^T p|}{\|p\|_2}. \tag{43}$$

Proof. We have

$$|\nabla f(u)^T p| = |F(u)^T J(u)p| \leq \|F(u)\|_2 \|J(u)\|_2 \|p\|_2. \tag{44}$$

Moreover, from $\|F(u)\|_2 = \|r - J(u)p\|_2 \leq \eta \|F(u)\|_2 + \|J(u)\|_2 \|p\|_2$ we get

$$\|F(u)\|_2 \leq \frac{\|J(u)\|_2}{(1-\eta)} \|p\|_2. \tag{45}$$

The result follows from combining (44) and (45). □

A consequence of the above lemma is that if $\|F(u_n) + J(u_n)p_n\|_2 \leq \eta \|F(u_n)\|_2$, then the backtracking procedure will always start with $\lambda = 1$ in the first step if ϵ^* is small enough, or to be more accurate as long as

$$\epsilon^* \leq \frac{1-\eta}{\|J(u_n)\|_2^2}. \tag{46}$$

We can now prove the following theorem.

THEOREM 3.10. *Let $f \equiv \frac{1}{2}\|F\|_2^2$ satisfy the conditions of Lemma* 3.8, *and let p_n be such that $\|F_n + J_n p_n\|_2 \leq \eta \|F_n\|_2$ for all n, with $0 \leq \eta < 1$. Furthermore, let each iterate be chosen by Algorithm* 3.1. *Then, either*

$$\lim_{n \to \infty} f(u_n) = 0 \tag{47}$$

or

$$\lim_{n \to \infty} \|p_n\|_2 = \infty. \tag{48}$$

Proof. Letting as before $r = F(u_n) + J(u_n)p_n$, and dropping the subscript n we have

$$\nabla f^T p = F^T(-F + r) \leq -\|F\|_2^2(1-\eta) = -2(1-\eta)f, \tag{49}$$

and as a result (with $u_{n+1} = u_n + \lambda_n p_n = u + \lambda p$),

$$f(u_{n+1}) \leq f(u) + \lambda \alpha \nabla f^T p \leq f(u) - 2\lambda\alpha(1-\eta)f(u) = f(u)[1 - 2\lambda\alpha(1-\eta)]. \tag{50}$$

From the result of Lemma 3.8 we have

$$-\lambda \|p\|_2 \leq \kappa \frac{\nabla f^T p}{\|p\|_2}, \tag{51}$$

where we define

$$\kappa \equiv \min\left\{\epsilon^*\ ,\ \frac{(1-\alpha)}{\gamma}\theta_{\min}\right\}$$

and therefore, (50) becomes

$$f(u_{n+1}) \leq f(u)\left[1 + 2\alpha\kappa(1-\eta)\frac{\nabla f^T p}{\|p\|_2^2}\right]. \tag{52}$$

Denoting by t_n the quantity $\nabla f_n^T p_n / \|p_n\|_2^2$ and by c the constant $2\alpha\kappa(1-\eta)$, we rewrite this relation as

$$f(u_{n+1}) \leq f(u_n)[1 + ct_n]. \tag{53}$$

Since $f(u_n)$ is bounded from below and nonincreasing, it converges to a certain limit ϕ. If this limit is zero the result of the theorem holds. If it is different from zero then by dividing both members of (53) by $f(u_n)$, we see that $1 + ct_n$, which is bounded from above by 1 and from below by a sequence converging to 1, has 1 as its limit. Equivalently, t_n converges to 0. Going back to the relation (49) which we rewrite as $2f(u_n)(1-\eta) \leq |t_n| \cdot \|p_n\|_2^2$, we see immediately that in this situation we must have $\|p_n\|_2 \to \infty$. □

Note that it is easy to adapt the proof of Lemma 3.8 to show a result of the form

$$\lambda_n \geq \min\left\{1, \frac{(1-\alpha)\theta_{\min}}{\gamma} \frac{\nabla f(u_n)^T p_n}{\|p_n\|_2^2}\right\}. \tag{54}$$

Hence, if we always take $\lambda = 1$ in step 1 of Algorithm 3.1 then we can use inequality (54) to show that

$$f(u_{n+1}) \leq f(u_n)\left[1 - 2\alpha(1-\eta)\min\left\{1, \frac{(1-\alpha)\theta_{\min}}{\gamma} \frac{\nabla f(u_n)^T p_n}{\|p_n\|_2^2}\right\}\right], \tag{55}$$

which can be exploited to establish the alternatives (47)–(48) with a minor modification of the above proof. This shows that it is sufficient to initialize $\lambda = 1$ in Algorithm 3.1 in order to show convergence, provided that the residual norm condition (30) is satisfied.

We mention that Eisenstat and Walker [12] have recently established an extension to Theorem 3.10. More precisely, they show that in addition to the conclusion of Theorem 3.10, one of the following holds:

(i) $\lim_{n\to\infty} \|u_n\|_2 = \infty$.

(ii) The sequence u_n has finite limit points, and F' is singular at each of them.

(iii) The sequence u_n converges to a limit u_* such that $F(u_*) = 0$ and $F'(u_*)$ is nonsingular.

If the sequence u_n does not verify (i) or (ii), then it must have a limit point u_* such that $J(u_*)$ is nonsingular. Since

$$\|p_n\|_2 \leq (1+\eta)\|J(u_n)^{-1}\|_2 \cdot \|F(u_n)\|_2$$

for u_n sufficiently close to u_*, condition (48) cannot hold and therefore by the theorem condition (47) holds. To prove that in this situation (iii) is true, Eisenstat and Walker use the following lemma to show that in fact $F(u_*) = 0$.

LEMMA 3.11 ([13, Thm. 3.2]). *Assume F is continuously differentiable, $\{u_k\}$ is a sequence such that $F(u_k) \to 0$, and for each k,*

$$\|F(u_{k+1})\| \leq \|F(u_k)\| \quad \text{and} \quad \|F(u_k) + J(u_k)s_k\| \leq \eta\|F(u_k)\|,$$

where $s_k = u_{k+1} - u_k$ and $\eta > 0$ is independent of k. If u_* is a limit point of $\{u_k\}$ such that $J(u_*)$ is nonsingular, then $F(u_*) = 0$ and $u_k \to u_*$. In the above lemma, the norm is arbitrary and the scalar η is not necessarily less than one.

We can show a result that is more explicit than Theorem 3.10 if we make a few additional assumptions on $J(u_n)$.

COROLLARY 3.12. *Let $f \equiv \frac{1}{2}\|F\|_2^2$ satisfy the conditions of Lemma 3.8 and let p_n be such that $\|F_n + J(u_n)p_n\|_2 \leq \eta\|F_n\|_2$ for each n, with $\eta < 1$. Furthermore, let each iterate be chosen by Algorithm 3.1, and assume that $J(u_n)^{-1}$ exists and its norm is bounded from above for all n. Then*

$$\lim_{n\to\infty} f(u_n) = 0. \tag{56}$$

Proof. From the relation (18), and the fact that $J(u_n)^{-1}$ is bounded from above, the norm of the vector $p_n = -J(u_n)^{-1}(F(u_n) - r)$ is bounded from above. Therefore, from the previous theorem, we must have $\lim_{n\to\infty} f(u_n) = 0$. □

The following additional results do not require the use of the backtracking procedure described in Algorithm 3.1. They are based upon the ideas presented by Dennis and Schnabel [11]. Given the current Newton iterate $u = u_n$ and a descent direction p, we want to take a step in the direction of p that yields an acceptable u_{n+1}. We will define a step $\delta = \lambda p$ to be *acceptable* if both of the Goldstein–Armijo [11] conditions are met, namely

$$f(u + \lambda p) \le f(u) + \alpha\lambda \nabla f(u)^T p \tag{57}$$

and

$$f(u + \lambda p) \ge f(u) + \beta\lambda \nabla f(u)^T p, \tag{58}$$

for given scalars α, β satisfying $0 < \alpha < \beta < 1$. Again, we will refer to these two conditions as the α and β conditions, respectively. For a given descent direction p, the next result shows that there exist points $u + \lambda p$ satisfying (57) and (58).

THEOREM 3.13. *Let $f : \mathbb{R}^N \to \mathbb{R}$ be continuously differentiable on $\mathbb{R}^N$ with $f(z) \ge 0$ for all $z \in \mathbb{R}^N$. Let $u, p \in \mathbb{R}^N$ be such that $\nabla f(u)^T p < 0$. Then, given $0 < \alpha < \beta < 1$, there exist $\lambda_u > \lambda_\ell > 0$ such that $u + \lambda p$ satisfies* (57) *and* (58) *for any $\lambda \in (\lambda_\ell, \lambda_u)$.*

This is essentially Theorem 6.3.2, in Dennis and Schnabel [11, p. 120], and so the proof is omitted.

THEOREM 3.14. *Let $f : \mathbb{R}^N \to \mathbb{R}$ be continuously differentiable on $\mathbb{R}^N$ with $f(z) \ge 0$ for all $z \in \mathbb{R}^N$, and assume there exists a constant $\gamma \ge 0$ such that*

$$\|\nabla f(z) - \nabla f(u)\|_2 \le \gamma \|z - u\|_2 \tag{59}$$

for every $u, z \in \mathbb{R}^N$. Then given any $u_0 \in \mathbb{R}^N$, there exists a sequence $\{u_n\}$ ($n = 0, 1, \ldots$) generated either from Algorithm 3.1 *or so as to satisfy the conditions* (57) *and* (58). *In addition this sequence is such that either*

$$\nabla f(u_n)^T \delta_n < 0$$

or

$$\nabla f(u_n) = 0 \quad \text{and} \quad \delta_n = 0,$$

for each $n > 0$, where $\delta_n \equiv u_{n+1} - u_n$. Furthermore, for any such sequence, either

$$\text{(a)} \qquad \nabla f(u_n) = 0 \quad \text{for some } n \ge 0, \text{ or}$$

$$\text{(b)} \qquad \lim_{n\to\infty} \frac{\nabla f(u_n)^T \delta_n}{\|\delta_n\|_2} = 0.$$

Proof. This is essentially Theorem 6.3.3 in [11, p. 121], except that condition (58) is slightly different and f is assumed to be bounded from below. For each n, if $\nabla f(u_n) = 0$, then (a) holds and the sequence is constant from then on. If $\nabla f(u_n) \ne 0$, then there exists a p_n such that $\nabla f(u_n)^T p_n < 0$ (e.g., take $p_n = -\nabla f(u_n)$). By Theorem 3.13, there exists $\lambda_n > 0$ such that $u_{n+1} = u_n + \lambda_n p_n$ satisfies (57) and (58).

Let $\delta_n = \lambda_n p_n$. We must now show that if no term of $\{\delta_n\}$ is zero, then (b) must hold.

First, define $\omega_n \equiv \|\delta_n\|_2$ and

$$\sigma_n \equiv \frac{\nabla f(u_n)^T \delta_n}{\omega_n}.$$

By (57) and $\omega_i \sigma_i < 0$ for every i, we have, for any $j > 0$,

$$f(u_{j-1}) - f(u_j) \geq -\alpha\, \omega_j \sigma_j > 0.$$

Since the sequence $f(u_j)$ is monotone decreasing, the sequence in the left-hand side must converge to zero and so $\omega_n \sigma_n \to 0$ as $n \to \infty$. To conclude that $\sigma_n \to 0$ we must use the relation (38) for sequences generated by Algorithm 3.1 or condition (58) for sequences generated from satisfying the α and β conditions. These conditions were used to guarantee that the steps do not get too small.

For sequences generated from the α and β condition, we use the mean value theorem to show that there exists a $\bar{\lambda} \in (0, \lambda_n)$ such that

$$f(u_{n+1}) = f(u_n) + \nabla f(u_n + \bar{\lambda} p_n)^T (u_{n+1} - u_n) \tag{60}$$

which, using condition (58), yields,

$$f(u_{n+1}) - f(u_n) = \nabla f(u_n + \bar{\lambda} p_n)^T (u_{n+1} - u_n) \geq \beta \nabla f(u_n)^T \delta_n. \tag{61}$$

This implies that

$$\left[\nabla f(u_n + \bar{\lambda} p_n) - \nabla f(u_n)\right]^T \delta_n \geq (\beta - 1) \nabla f(u_n)^T \delta_n > 0.$$

Therefore,

$$\begin{aligned} 0 < (\beta - 1)\omega_n \sigma_n &\leq \omega_n \|\nabla f(u_n + \bar{\lambda} p_n) - \nabla f(u_n)\|_2 \\ &\leq \gamma \bar{\lambda} \|p_n\|_2 \omega_n \leq \gamma \omega_n^2. \end{aligned}$$

So,

$$\omega_n \geq \frac{\beta - 1}{\gamma} \sigma_n > 0 \tag{62}$$

and

$$\omega_n \sigma_n \leq \frac{\beta - 1}{\gamma} \sigma_n^2 < 0.$$

Hence, $\omega_n \sigma_n \to 0$ implies $\sigma_n \to 0$ as $n \to \infty$.

For sequences generated by Algorithm 3.1, the relation (38) implies immediately that

$$\omega_n \geq -\min\left\{\epsilon^*, \frac{1 - \alpha}{\gamma} \theta_{\min}\right\} \sigma_n$$

which is analogous to (62), and the proof can be completed in the same manner. □

Note that $\sigma_n \equiv \nabla f(u_n)^T \delta_n / \omega_n \to 0$ does not imply that $\nabla f(u_n) \to 0$ as $n \to \infty$. However, it will as long as the angle between $\nabla f(u_n)$ and δ_n is bounded away from

90°. It is possible to guarantee that this is the case in the inexact Newton setting, with $f \equiv \frac{1}{2}\|F\|_2^2$, as was shown by Lemma 3.4. Note also that $\nabla f_n \to 0$ does not imply $F(u_n) \to 0$, without some additional assumptions, e.g., as in Corollary 3.12.

If conclusion (b) holds in Theorem 3.14 with $f \equiv \frac{1}{2}\|F\|_2^2$ and $\|F(u_n)+J(u_n)\delta_n\|_2 \leq \eta\|F(u_n)\|_2$ for all n, where $\eta \in (0,1)$ and if we assume that the condition numbers $M_n = \text{cond}_2(J(u_n))$ are uniformly bounded from above, then (17) shows that $\sigma_n \to 0$ as $n \to \infty$ does imply $\nabla f(u_n) \to 0$. However, the conditions are too weak to imply that $\{u_n\}$ converges. On the other hand, one can exploit the result proved by Eisenstat and Walker cited earlier in the following manner. If $\|u_n\|_2 \not\to \infty$, then $\{u_n\}$ must have a limit point u_*. Since $\{M_n\}$ is uniformly bounded from above, $J(u_*)$ is nonsingular; furthermore, since $\nabla f(u_n) = J(u_n)^T F(u_n) \to 0$ and $\|F(u_n)\|_2$ is monotone decreasing, it follows that $F(u_n) \to 0$. Since $\|F(u_{n+1})\|_2 \leq \|F(u_n)\|_2$ and $\|F(u_n) + J(u_n)\delta_n\|_2 \leq \eta\|F(u_n)\|_2$, Lemma 3.11 establishes that $F(u_*) = 0$ and $u_n \to u_*$.

3.2. Superlinear convergence of inexact Newton sequences. In this subsection we will require that $\eta_n \to 0$ as $n \to \infty$. As noted in Theorem 2.1, given that the sequence of inexact Newton iterates converges, this additional assumption on the η_n's implies that the convergence is at least superlinear. The main result of this subsection is a modification of a theorem obtained by Dennis and Moré [10], and shows that the global strategy based on the above α and β conditions will permit full inexact Newton steps when close to a minimizer of f, provided that $\alpha < \frac{1}{2}$ and $\beta > \frac{1}{2}$.

THEOREM 3.15. *Let $f \equiv \frac{1}{2}\|F\|_2^2$ satisfy the conditions of Theorem 3.14. In addition, let F be twice continuously differentiable in an open convex set $D \subset \mathbb{R}^N$, and assume $\nabla^2 f \in Lip_\gamma(D)$. Consider a sequence $\{u_n\}$ generated by $u_{n+1} = u_n + \lambda_n p_n$, where $\|F(u_n) + J(u_n)p_n\|_2 \leq \eta_n\|F(u_n)\|_2$ for all n with $0 < \eta_n \leq \eta < 1$ for all n, and λ_n chosen so that (57) and (58) hold with $\alpha < \frac{1}{2}$ and $\beta > \frac{1}{2}$.*

If $u_n \to u_* \in D$ with $J(u_*)$ nonsingular, then $F(u_*) = 0$. If in addition, $\eta_n \to 0$ as $n \to \infty$, then there exists an $n_0 \geq 0$ such that for all $n \geq n_0$, $\lambda_n = 1$ is admissible (i.e., satisfies conditions (57) and (58)). Furthermore, if $\lambda_n = 1$ for all $n \geq n_0$, then $u_n \to u_*$ superlinearly. If also $\eta_n = O(\|F(u_n)\|_2)$, then the convergence is quadratic.

Proof. If $u_n \to u_* \in D$ with $J(u_*)$ nonsingular, then $J(u_n)$ is nonsingular for $n \geq n_1$, with n_1 large enough. Next, if $F(u_n) = 0$ for some $n \geq n_1$, then the residual condition implies p_n must be zero. As a result, $u_{n'} = u_n$ for all $n' \geq n$, and the result holds. Therefore, without loss of generality we may assume that $F(u_n)$ is nonzero and $J(u_n)$ is nonsingular for all n.

Since $f(u) = \frac{1}{2}F(u)^T F(u) \geq 0$, and since $\|F_n + J_n p_n\|_2 \leq \eta\|F_n\|_2$ (where $F_n = F(u_n)$, $J_n = J(u_n)$, etc.) implies $\nabla f_n^T p_n < 0$ for all n, with $\eta \in (0,1)$, we have by Theorem 3.14 that

$$\lim_{n\to\infty} \frac{\nabla f_n^T p_n}{\|p_n\|_2} = 0. \tag{63}$$

If $u_n \to u_* \in D$ with $J_* = J(u_*)$ nonsingular, then by continuity $M_n = \text{cond}_2(J_n) \to M_* = \text{cond}_2(J_*)$, and so the sequence $\{M_n\}$ is uniformly bounded from above. Thus, the discussion after the proof of Lemma 3.4 implies that $\nabla f(u_n) \to 0$ as $n \to \infty$. Hence, again by continuity

$$0 = \nabla f(u_*) = J_*^T F_*,$$

which gives $F(u_*) = 0$.

Next, we show that $\|p_n\|_2 \to 0$ as $n \to \infty$. Since

$$p_n = -J_n^{-1}F_n + J_n^{-1}(F_n + J_n p_n),$$

we have

$$\begin{aligned} \|p_n\|_2 &\le \|J_n^{-1}\|_2 \cdot \|F_n\|_2 + \|J_n^{-1}\|_2 \cdot \|F_n + J_n p_n\|_2 \\ &\le (1+\eta)\|J_n^{-1}\|_2 \cdot \|F_n\|_2. \end{aligned} \tag{64}$$

Thus, $\|F_n\|_2 \to 0$ implies $\|p_n\|_2 \to 0$ as $n \to \infty$. Also,

$$\|\nabla f_n\|_2 = \|J_n^T F_n\|_2 \ge \|J_n^T\|_2^{-1} \cdot \|F_n\|_2 = \|J_n\|_2^{-1} \cdot \|F_n\|_2.$$

Hence, by Lemma 3.4 we have

$$\frac{-\nabla f_n^T p_n}{\|p_n\|_2} \ge \frac{1-\eta}{(1+\eta)M_n}\|J_n\|_2^{-1} \cdot \|F_n\|_2,$$

where $M_n = \mathrm{cond}_2(J_n)$. Thus,

$$\|F_n\|_2 \le \frac{(1+\eta)M_n}{1-\eta}\|J_n\|_2 \cdot \frac{-\nabla f_n^T p_n}{\|p_n\|_2},$$

and so

$$\|F_n\|_2 \cdot \|p_n\|_2 \le \frac{(1+\eta)M_n}{1-\eta}\|J_n\|_2 \cdot (-\nabla f_n^T p_n) = -a_n \nabla f_n^T p_n, \tag{65}$$

letting $a_n = ((1+\eta)/(1-\eta))M_n \cdot \|J_n\|_2$. In addition, it immediately follows from (64) that

$$\|p_n\|_2^2 \le \frac{(1+\eta)^2}{1-\eta}M_n^2(-\nabla f_n^T p_n) = -b_n \nabla f_n^T p_n, \tag{66}$$

letting $b_n = ((1+\eta)^2/(1-\eta))M_n^2$.

Using (65) and (66), and the fact that $\|p_n\|_2 \to 0$, we next show that $\lambda_n = 1$ satisfies conditions (57) and (58) for n large. First note that if $F = (F_1, \ldots, F_N)^T$, then

$$\begin{aligned} \nabla^2 f(u) &= J(u)^T J(u) + \sum_{i=1}^{N} F_i(u)\nabla^2 F_i(u) \\ &\equiv J(u)^T J(u) + S(u). \end{aligned}$$

Also note that $\|S(u_n)\|_2 \to 0$ as $n \to \infty$ since $u_n \to u_*$ and $F(u_*) = 0$. For each n, by the mean value theorem there exists a $\bar{u}_n$ on the line segment between u_n and $u_n + p_n$ such that

$$f(u_n + p_n) - f_n - \tfrac{1}{2}\nabla f_n^T p_n = \tfrac{1}{2}\left(\nabla f_n + \nabla^2 f(\bar{u}_n)p_n\right)^T p_n.$$

This gives

$$\begin{aligned}
\left| f(u_n + p_n) - f_n - \frac{1}{2}\nabla f_n^T p_n \right| &= \frac{1}{2}\left| \left(\nabla f_n + \nabla^2 f(\bar{u}_n)p_n\right)^T p_n \right| \\
&= \left| \frac{1}{2}\left(\nabla f_n + \nabla^2 f_n p_n\right)^T p_n + \frac{1}{2}p_n^T\left(\nabla^2 f(\bar{u}_n) - \nabla^2 f_n\right)p_n \right| \\
&\leq \frac{1}{2}\left(\|J_n^T(F_n + J_n p_n)\|_2 \cdot \|p_n\|_2 + (\|S_n\|_2 + \gamma\|p_n\|_2)\|p_n\|_2^2\right) \\
&\leq \frac{1}{2}\left(\eta_n\|J_n\|_2 \cdot \|F_n\|_2 \cdot \|p_n\|_2 + (\|S_n\|_2 + \gamma\|p_n\|_2)\|p_n\|_2^2\right) \\
&\leq -\frac{1}{2}\left(a_n\eta_n\|J_n\|_2 + b_n(\|S_n\|_2 + \gamma\|p_n\|_2)\right)\nabla f_n^T p_n \\
&= -\frac{1}{2}\epsilon_n \nabla f_n^T p_n,
\end{aligned}$$

where we have used (65) and (66), and defined $\epsilon_n \equiv a_n\eta_n\|J_n\|_2 + b_n(\|S_n\|_2 + \gamma\|p_n\|_2)$. Therefore,

$$\tfrac{1}{2}(1+\epsilon_n)\nabla f_n^T p_n \leq f(u_n + p_n) - f_n \leq \tfrac{1}{2}(1-\epsilon_n)\nabla f_n^T p_n.$$

Next, note that since a_n, b_n and $\|J_n\|_2$ are all bounded from above, and since η_n, $\|S_n\|_2$ and $\|p_n\|_2$ all converge to 0 as $n \to \infty$, we have that $\epsilon_n \to 0$.

We need to choose $n_0 \geq 0$ so that for all $n \geq n_0$ we have

$$\epsilon_n \leq \min\{1 - 2\alpha, 2\beta - 1\}.$$

It then follows that for all $n \geq n_0$,

$$\beta\nabla f_n^T p_n \leq f(u_n + p_n) - f_n \leq \alpha\nabla f_n^T p_n.$$

Thus, $\lambda_n = 1$ is admissible for all $n \geq n_0$.The superlinear (quadratic) convergence of the sequence follows from Corollary 3.5 in [9] or Theorem 2.1 in this paper. □

In a manner similar to the remark at the end of the previous section, one can apply Lemma 3.11 to give a stronger statement of the conclusions in Theorem 3.15. Namely, one can relax the statement to the following: *If $\{u_n\}$ has a limit point u_* such that $J(u_*)$ is nonsingular, then $F(u_*) = 0$ and $u_n \to u_*$. If in addition, $u_* \in D$ and $\eta_n \to 0$ as $n \to \infty$, then ....* An analogous restatement applies to Corollary 3.16 below.

For sequences generated by Algorithm 3.1, the above proof also shows that $\lambda_n = 1$ will be admissible for n large enough, with one possible exception. More precisely, if $\lambda_n = 1$ is presented to the second step of the algorithm, then it will pass the α condition test and be accepted. This is because the above proof establishes that for n large enough both the α and β conditions are satisfied. The possible exception to this comes from the very first step of Algorithm 3.1 which may prevent λ_n from ever taking the value one. However, as was noted earlier, the initial λ_n in the procedure can be made to take the value 1, provided we are able to select ϵ^* to satisfy (46) for all iterates. In particular, we can say that if $\|J(u_n)\|_2$ is bounded from above by M for all n, and ϵ^* is small enough, i.e., such that

$$\epsilon^* \leq \frac{1-\eta}{M^2},$$

then the above theorem also holds for sequences generated by Algorithm 3.1, in which the directions p_n satisfy the same residual norm restrictions as those in the theorem. We note, though, that such a condition on ϵ^* is not practical. (As noted earlier, if p_n is always required to satisfy a residual norm condition (30), then taking the initial $\lambda_n = 1$ is okay at each step.)

One can relax the condition that $\eta_n \to 0$ in the above theorem somewhat, although the resulting condition on η is not computationally feasible in general.

COROLLARY 3.16. *Let F and $f \equiv \frac{1}{2}F^TF$ satisfy the conditions of Theorem* 3.15. *Assume that $M = \sup_{u\in D}\{\text{cond}_2(J(u))\} < \infty$ and that $K = \sup_{u\in D}\{\|J(u)\|_2\} < \infty$. Consider a sequence $\{u_n\}$ generated by $u_{n+1} = u_n + \lambda_n p_n$, where $\|F(u_n) + J(u_n)p_n\|_2 \le \eta\|F(u_n)\|_2$ for all n with $0 < \eta < 1$ for all n, and λ_n chosen so that* (57) *and* (58) *hold with $\alpha < \frac{1}{2}$ and $\beta > \frac{1}{2}$.*

If $u_n \to u_* \in D$ with $J(u_*)$ nonsingular, then $F(u_*) = 0$. If in addition, η satisfies

$$\eta\frac{1+\eta}{1-\eta} \cdot M \cdot K < \min\{1-2\alpha, 2\beta-1\}, \tag{67}$$

then there exists an $n_0 \ge 0$ such that for all $n \ge n_0$, $\lambda_n = 1$ is admissible (i.e., satisfies conditions (57) *and* (58)). *Furthermore, if $\lambda_n = 1$ for all $n \ge n_0$, then $u_n \to u_*$ linearly.*

Proof. From the proof of Theorem 3.15, the corollary will be true if we have

$$a_n\eta\|J_n\|_2 < \min\{1-2\alpha, 2\beta-1\},$$

for all sufficiently large n. But this follows immediately from the definitions of M and K, and condition (67). □

One may wonder whether or not the conditions required in the theorem are too strong if we want to ensure that $\lambda_n = 1$ is admissible. More precisely, does the weaker condition $\|F_n + J_np_n\|_2 \le \eta\|F_n\|_2$ for all n, where $\eta \in (0,1)$, imply the admissibility of $\lambda_n = 1$ for all large n whenever a sequence $u_{n+1} = u_n + \lambda_n p_n$ converges to a solution u_*? The answer is no, as shown by the following example.

Example. Consider the one-dimensional function $F(u) = u \in \mathbb{R}$. Choose $\eta \in (0,1)$ and $0 < \alpha < \frac{1}{2} < \beta < 1$ so that

$$2\beta - 1 < \eta.$$

Choose θ so that $0 < \theta < 1$ and

$$2\beta - 1 < 1 - \theta < \eta.$$

Consider a sequence $\{u_n\}$ generated by $u_{n+1} = u_n + \lambda_n p_n$, where $u_0 = 1$, $p_n = -\theta u_n$, and $\lambda_n = (2-\alpha-\beta)/\theta$ for all n. Then $f(u) = \frac{1}{2}u^2$ with $\nabla f(u_n)^T p_n = -\theta u_n^2$, and so p_n is a descent direction for all n. Also, $\|F_n + J_np_n\|_2 = |(1-\theta)u_n| \le \eta\|F(u_n)\|_2 = \eta|u_n|$ for all n. We next show that λ_n is admissible for all n. We have

$$f(u_n + \lambda p_n) = \tfrac{1}{2}(u_n + \lambda p_n)^2 = \tfrac{1}{2}u_n^2 - \lambda\theta u_n^2 + \tfrac{1}{2}\lambda^2\theta^2 u_n^2.$$

Next,

$$f(u_n) + \lambda\alpha\nabla f(u_n)^T p_n = \tfrac{1}{2}u_n^2 + \lambda\alpha(-\theta u_n^2),$$

and

$$f(u_n) + \lambda\beta\nabla f(u_n)^T p_n = \tfrac{1}{2}u_n^2 + \lambda\beta(-\theta u_n^2).$$

Thus, λ is admissible if and only if

$$\tfrac{1}{2}u_n^2 + \lambda\beta(-\theta u_n^2) < \tfrac{1}{2}u_n^2 - \lambda\theta u_n^2 + \tfrac{1}{2}\lambda^2\theta^2 u_n^2 < \tfrac{1}{2}u_n^2 + \lambda\alpha(-\theta u_n^2),$$

or if and only if

$$\frac{2(1-\beta)}{\theta} < \lambda < \frac{2(1-\alpha)}{\theta}.$$

Clearly, the λ_n defined above is admissible for all n. However, for the parameters given above we have $1 < 2(1-\beta)/\theta$, and so $\lambda = 1$ can never satisfy conditions (57) and (58). Notice that $u_{n+1} = (\alpha + \beta - 1)u_n$, and so $u_n = (\alpha + \beta - 1)^n u_0$ with $|\alpha + \beta - 1| < 1$. Hence, $u_n \to 0$ as $n \to 0$ linearly and, if $\alpha + \beta \neq 1$, no faster.

Note that the convergence results of this section are similar to others in the literature. However the emphasis was put on the additional residual norm condition (30). As was seen, slightly different, and somewhat stronger, results can be shown in the situation where δ_n satisfies a residual norm condition.

4. Convergence results for model trust region techniques. In [5], we presented a trust region algorithm based on a Newton–GMRES iteration. We give here a convergence theory for this algorithm and other methods based on projection principles. We start by describing in §4.1 general trust region methods and give some background on their theory. Our basic approach will be modelled after the work of Shultz, Schnabel, and Byrd [29]. In §4.2 we will adapt this theory to the particular case that is of interest to us, namely the case in which a projection method onto a lower-dimensional subspace is used.

4.1. General trust region techniques for nonlinear optimization. The model trust region algorithm generates a sequence of points u_n, and at the nth stage of the iteration a quadratic model of $f : \mathbb{R}^N \to \mathbb{R}$ near the current iterate u_n is used which has the form

$$\psi_n(w) = f_n + g_n^T w + \tfrac{1}{2}w^T B_n w,$$

where $f_n = f(u_n)$, $g_n = \nabla f(u_n)$, and $B_n \approx \nabla^2 f(u_n)$. (A projected analogue of the above function based on approximations from a subspace K will be developed later.) At this stage an initial value for the trust region size τ_n is also available. An inner iteration is then performed which consists of using the current trust region size τ_n and the information contained in the quadratic model to compute a step

$$p_n(\tau_n) = p(g_n, B_n, \tau_n).$$

Then a comparison of the actual reduction of the objective function

$$\text{ared}_n(\tau_n) = f_n - f(u_n + p_n(\tau_n))$$

and the reduction predicted by the quadratic model

$$\text{pred}_n(\tau_n) = f_n - \psi_n(p_n(\tau_n)),$$

is performed. If there is a satisfactory reduction, then the step can be taken, or a possibly larger trust region used. If not, then the trust region is reduced and the inner iteration is repeated.

For now we leave unspecified what algorithm is used to form the step computing function $p(g, B, \tau)$, and how the trust region size or radius τ_n is changed. We also leave unspecified the selection of B_n except to restrict it to be symmetric positive definite. Details on these options will be given later. Shultz, Schnabel, and Byrd [29] describe an abstract trust region algorithm as follows.

ALGORITHM 4.1. General Trust Region Algorithm.

1. Choose $\gamma_1, \alpha_1, \alpha_2 \in (0,1)$, $u_1 \in \mathbb{R}^N$, $\tau_1 > 0$, and let $n = 1$.
2. Compute $f_n = f(u_n)$, $g_n = g(u_n) \equiv \nabla f(u_n)$, and $B_n \in \mathbb{R}^{N \times N}$ symmetric and positive definite.
3. Find τ_n and compute $p_n = p_n(\tau_n)$ satisfying: $\|p_n\|_2 \leq \tau_n$ and
   (a) $\text{ared}_n(\tau_n)/\text{pred}_n(\tau_n) \geq \alpha_1$ and
   (b) either $\tau_n \geq \tau_{n-1}$, or
      - $\tau_n \geq \|B_{n-1}^{-1} g_{n-1}\|_2$, or
      - for some $\tau \leq \tau_n/\gamma_1$, $\text{ared}_n(\tau)/\text{pred}_n(\tau) < \alpha_2$ or $\text{ared}_{n-1}(\tau)/\text{pred}_{n-1}(\tau) < \alpha_2$.
4. Let $u_{n+1} = u_n + p_n$ and $n = n + 1$.
5. Go to Step 2.

The conditions that the step selection function $p(g, B, \tau)$ must satisfy will now be considered. Defining

$$\text{pred}(g, B, \tau) = -g^T p(g, B, \tau) - \tfrac{1}{2} p(g, B, \tau)^T B p(g, B, \tau),$$

the conditions that we consider follow.

Condition 1. There exist two scalars $\bar{c}_1, \sigma_1 > 0$ such that for all $g \in \mathbb{R}^N$, for all symmetric positive definite $B \in \mathbb{R}^{N \times N}$, and for all $\tau > 0$, $\text{pred}(g, B, \tau) \geq \bar{c}_1 \|g\|_2 \min\{\tau, \sigma_1 \|g\|_2 / \|B\|_2\}$.

Condition 2. If B is symmetric positive definite and $\|B^{-1} g\|_2 \leq \tau$, then $p(g, B, \tau) = -B^{-1} g$.

The first condition requires that the predicted decrease be at least as large as a given multiple of the minimum decrease that would be provided by a quadratic search along the steepest descent direction. The second condition forces the direction p to be equal to the Newton direction whenever the next point $u_n + p$ lies in the trust region. The following result is given in [29].

THEOREM 4.1. *Let $f : \mathbb{R}^N \to \mathbb{R}$ be twice continuously differentiable and bounded below, and let $\nabla^2 f(u)$ satisfy $\|\nabla^2 f(u)\|_2 \leq \beta_1$ for all $u \in \mathbb{R}^N$. Suppose that an algorithm satisfying the conditions of Algorithm* 4.1 *above is applied to $f(u)$, starting from some $u_1 \in \mathbb{R}^N$, generating a sequence $\{u_n\}$, $n = 1, 2, \ldots$. Then we have the following.*

(i) *If $p(g, B, \tau)$ satisfies Condition* 1 *with $\|B_n\|_2 \leq \beta_2$ for all n, then $g_n \to 0$.*

(ii) *If $p(g, B, \tau)$ satisfies Conditions* 1 *and* 2, *$B_n = \nabla^2 f(u_n)$ for all n, $\nabla^2 f(u)$ is Lipschitz continuous with constant L, and u_* is a limit point of $\{u_n\}$ with $\nabla^2 f(u_*)$ positive definite, then u_n converges to u_* q-quadratically.*

The first result in the theorem gives the first order stationary point convergence of the sequence of iterates while the first and second results taken together give second order stationary point convergence.

We next discuss the procedures normally used to form the step computing function $p(g, B, \tau)$. One way in which this can be done is to take $p(g, B, \tau)$ to be the solution of the minimization problem

$$\min_{\|w\|_2 \leq \tau} \psi(w), \quad \text{where } \psi(w) = f + g^T w + \tfrac{1}{2} w^T B w.$$

Assuming B is symmetric and positive definite, the solution to this problem is given by

$$\bar{p}(\tau) = \begin{cases} -(B + \mu I)^{-1} g, & \text{with } \|(B + \mu I)^{-1} g\|_2 = \tau \text{ when } \|B^{-1} g\|_2 > \tau, \\ -B^{-1} g, & \text{when } \|B^{-1} g\|_2 \leq \tau. \end{cases}$$

We note that the μ in the first part of the definition of $\bar{p}$ is unique. It is well known that for $p(g, B, \tau) = \bar{p}(\tau)$ the following inequality is true:

$$f - \psi(\bar{p}(\tau)) \geq \frac{1}{2} \|g\|_2 \min \left\{ \tau, \frac{\|g\|_2}{\|B\|_2} \right\}.$$

For a proof of this result, see [19]. With this ideal choice of the direction $p(g, B, \tau)$, Condition 1 is trivially satisfied.

Since there is no finite method of determining μ such that $\|(B + \mu I)^{-1} g\|_2 = \tau$ when $\tau < \|B^{-1} g\|_2$, a piecewise linear approximation to $\bar{p}(\tau)$ frequently is used. The *dogleg strategy* of Powell [25] is an example of such a procedure. (See [11], [18] for a discussion of this and other dogleg strategies.) If we denote Powell's dogleg solution by $\hat{p}(\tau)$, then it is also well known that for $p(g, B, \tau) = \hat{p}(\tau)$ the following lemma is true.

LEMMA 4.2. *Let $\hat{p}(\tau)$ be the dogleg solution to the minimization problem*

$$\min_{\|w\|_2 \leq \tau} \psi(w), \tag{68}$$

where $\psi(w) = f + g^T w + \frac{1}{2} w^T B w$ with B symmetric and positive-definite. Then

$$f - \psi(\hat{p}(\tau)) \geq \frac{1}{2} \|g\|_2 \min \left\{ \tau, \frac{\|g\|_2}{\|B\|_2} \right\}. \tag{69}$$

For a proof of this lemma, see Powell [26]. Again, a consequence of the above lemma is that Condition 1 is trivially satisfied, and as a result of Theorem 4.1 global convergence will take place under the mild condition that $\|B_n\|_2$ remains bounded.

4.2. Application to projection methods for nonlinear equations. In the context of nonlinear equations, one typically bases the global strategy on the related function $f(u) \equiv \frac{1}{2} F(u)^T F(u)$. Letting u be the current approximate solution, the local quadratic model now has the form

$$\psi(w) = f + g^T w + \tfrac{1}{2} w^T B w,$$

where $f = f(u)$, $g = \nabla f(u) = J(u)^T F(u)$, and B approximates $J(u)^T J(u)$. Note that $\nabla^2 f(u) = J(u)^T J(u) + \sum_{i=1}^{N} F_i(u) \nabla^2 F_i(u)$, and so in general $\nabla^2 f(u) = J(u)^T J(u)$ only when $F(u) = 0$. When using projection methods to solve the nonlinear system, the full quadratic model $\psi(w)$ is replaced by a quadratic model on a lower-dimensional

subspace K. Letting the columns of the N x m matrix V form an orthonormal basis for K (with m the dimension of K), we have

$$\begin{aligned}\phi(y) &\equiv \psi(Vy) \\ &= f + g^T Vy + \tfrac{1}{2}(Vy)^T BVy \\ &= f + (V^T g)^T y + \tfrac{1}{2} y^T (V^T BV) y.\end{aligned}$$

Thus, $\nabla\phi(0) = V^T g$. Note that since V has orthonormal columns, the matrix $V^T BV$ is symmetric positive definite whenever B is the same. In the current setting, we will take $B = J(u)^T J(u)$. If $\hat{q}(\tau)$ is the dogleg solution to the minimization problem

$$\min_{\|y\|_2 \le \tau} \phi(y),$$

then Lemma 4.2 implies

$$f - \phi(\hat{q}(\tau)) \ge \frac{1}{2}\|V^T g\|_2 \min\left\{\tau, \frac{\|V^T g\|_2}{\|V^T BV\|_2}\right\}.$$

We then have

$$f - \phi(\hat{q}(\tau)) \ge \frac{1}{2}\|V^T g\|_2 \min\left\{\tau, \frac{\|V^T g\|_2}{\|B\|_2}\right\}, \tag{70}$$

using the fact that $\|V^T BV\|_2 \le \|B\|_2$.

In order to be able to apply the results of Shultz, Schnabel, and Byrd [29], we must convert the lower bound in the above inequality to one involving $\|g\|_2$, and not $\|V^T g\|_2$. As indicated above, we have $f = \frac{1}{2}F^T F$, and so $\psi(w)$ has the form

$$\psi(w) = f + (J^T F)^T w + \tfrac{1}{2} w^T (J^T J) w$$

since $B = J^T J$. Thus, we need a lower bound for $\|V^T g\|_2 = \|V^T J^T F\|_2$. Such a lower bound was derived in §3. Indeed, Corollary 3.5 states that

$$\|V^T \nabla f\|_2 = \|V^T g\|_2 \ge \frac{1-\eta}{(1+\eta)M}\|g\|_2, \tag{71}$$

where $M = \mathrm{cond}_2(J)$ with $\eta < 1$, provided that there exists at least one vector p in K such that

$$\|F(u) + J(u)p\|_2 \le \eta\|F(u)\|_2. \tag{72}$$

Therefore, we immediately have the following lemma.

LEMMA 4.3. *Let $J \in \mathbb{R}^{N \times N}$ be nonsingular and $F \in \mathbb{R}^N$ be given. Let $\eta \in [0,1)$ be chosen and let K be a subspace of dimension m in $\mathbb{R}^N$ such that*

$$\min_{p \in K} \|F + Jp\|_2 \le \eta\|F\|_2.$$

Choose the N x m matrix V so that its columns form an orthonormal basis for K. Let $\hat{q}(\tau)$ be the dogleg solution to

$$\min_{\|y\|_2 \le \tau} \phi(y), \quad \textit{where } \phi(y) = f + (V^T g)^T y + \tfrac{1}{2} y^T (V^T BV) y,$$

with $f = \frac{1}{2}F^TF$, $g = J^TF$ *and* $B = J^TJ$. *Then*

$$\|V^Tg\|_2 \geq \sigma\|g\|_2, \tag{73}$$

and

$$f - \phi(\hat{q}(\tau)) \geq \frac{1}{2}\sigma\|g\|_2 \min\left\{\tau, \sigma\frac{\|g\|_2}{\|B\|_2}\right\}, \tag{74}$$

where

$$\sigma \equiv \frac{1-\eta}{(1+\eta)M}, \tag{75}$$

with $M = \text{cond}_2(J)$.

Given this lemma, it is now possible to state a model trust region algorithm appropriate for use with a general projection method. The algorithm is stated in terms of a sequence of general subspaces K_n; also, recall that $B_n \equiv J_n^TJ_n$.

ALGORITHM 4.2. Inexact Newton Trust Region Algorithm.

1. Choose an $\eta_{\max} \in (0,1)$.
2. Choose $\gamma_1, \alpha_1, \alpha_2 \in (0,1)$, $u_1 \in \mathbb{R}^N$, $\tau_1 > 0$, and let $n = 1$.
3. Compute $F_n = F(u_n)$ and choose $\eta_n \in [0, \eta_{\max})$. Then choose a subspace $K_n \subset \mathbb{R}^N$ satisfying
   - $\min_{p\in K_n} \|F_n + J(u_n)p\|_2 \leq \eta_n\|F_n\|_2$.

   Let m_n be the dimension of K_n, and build $V_n \in \mathbb{R}^{N \times m_n}$ whose columns form an orthonormal basis for K_n.
4. Compute $f_n = \frac{1}{2}F_n^TF_n$, $V_n^Tg_n = V_n^TJ(u_n)^TF_n$, and $\bar{B}_n \equiv V_n^TJ(u_n)^TJ(u_n)V_n = V_n^TB_nV_n$.
5. Find τ_n and compute $p_n = V_nq_n = V_nq_n(\tau_n)$ satisfying $\|p_n\|_2 \leq \tau_n$ and
   (a) $\text{ared}_n(\tau_n)/\text{pred}_n(\tau_n) \geq \alpha_1$ and
   (b) either $\tau_n \geq \tau_{n-1}$, or
      - $\tau_n \geq \|\bar{B}_{n-1}^{-1}V_{n-1}^Tg_{n-1}\|_2$, or
      - for some $\tau \leq \tau_n/\gamma_1$, $\text{ared}_n(\tau)/\text{pred}_n(\tau) < \alpha_2$ or $\text{ared}_{n-1}(\tau)/\text{pred}_{n-1}(\tau) < \alpha_2$.
6. Let $u_{n+1} = u_n + p_n$ and $n = n + 1$.
7. Go to Step 3.

We note that when using the Arnoldi or GMRES algorithm to generate the subspace K_n the matrix $J(u_n)$ never needs to be calculated explicitly. All that is needed is the action of $J(u_n)$ times an arbitrary vector, and this can be approximated using difference quotients as was mentioned earlier. In addition, while not requiring $J(u_n)$ nonsingular in the trust region algorithm, we are allowing for the possibility of a breakdown in step 3 since the underlying linear system $J(u_n)p = -F_n$ may not have a solution.

The step selection function $q_n(V_n^Tg_n, \bar{B}_n, \tau_n)$ in Algorithm 4.2 is given by

$$q_n(V_n^Tg_n, \bar{B}_n, \tau_n) = \hat{q}_n(\tau_n),$$

where $\hat{q}_n(\tau_n)$ is the dogleg solution to the minimization problem

$$\min_{\|y\|_2\leq\tau_n} \phi_n(y) \quad \text{with } \phi_n(y) \equiv \psi_n(V_ny) = f_n + (V_n^Tg_n)^Ty + \tfrac{1}{2}y^T\bar{B}_ny.$$

This choice of step selection function was made for simplicity. Other multisegmented dogleg paths are also possible choices. If $J_n = J(u_n)$ is nonsingular, then by Lemma 4.3 we have

$$f_n - \phi_n(q_n) \geq \frac{1}{2}\sigma_n \|g_n\|_2 \min\left\{\tau_n, \frac{\sigma_n \|g_n\|_2}{\|B_n\|_2}\right\}, \tag{76}$$

where $B_n = J_n^T J_n$ and

$$\sigma_n = \frac{1 - \eta_n}{(1 + \eta_n) M_n}$$

with $M_n = \text{cond}_2(J_n)$. We now state the main result of this section.

THEOREM 4.4. *Let $F : \mathbb{R}^N \to \mathbb{R}^N$ be twice continuously differentiable. Define $f = \frac{1}{2}F^T F$ and assume $\|\nabla^2 f(u)\|_2 \leq \beta_1$ for all $u \in \mathbb{R}^N$. Suppose that an algorithm satisfying the conditions of Algorithm 4.2 above is applied to $f(u)$, starting from some $u_1 \in \mathbb{R}^N$, generating a sequence $\{u_n\}$, and assume that $\|B_n\|_2 \leq \beta_2$ and $\text{cond}_2(J_n) \leq M$ for all n. Then there is the following.*

(i) $g_n \to 0$.

(ii) *If $\nabla^2 f(u)$ is Lipschitz continuous with constant L, and u_* is a limit point of $\{u_n\}$ with $J(u_*)$ nonsingular, then $F(u_*) = 0$ and $u_n \to u_*$. If in addition, the sequence $\eta_n \to 0$ as $n \to \infty$, then u_n converges to u_* superlinearly. Also, if $\eta_n = O(\|F(u_n)\|_2)$, then the convergence is quadratic.*

This theorem is a direct adaptation of Theorem 4.1 above and the results of [9]. The proof uses the previous lemma, and due to its length is deferred to an appendix.

5. Applications to Newton–Krylov methods. In this section we show how the theory outlined in the previous sections applies to the Newton–Krylov methods discussed in §2. The details of the implementation of the Newton–Krylov algorithms when combined with either a linesearch backtracking strategy or model trust region technique are discussed at length in our earlier paper [5], and so we will not go into them here. As was seen above, the most important condition to guarantee convergence is that the residual norm be reduced by a certain amount. This was crucial in both the linesearch methods and the trust region methods. Unfortunately, as we indicate below, there is always the possibility of stagnation when using either linear Arnoldi or GMRES, and as a result there remains the possibility of a breakdown in the nonlinear iteration. In these situations the basic residual condition (4) may not be satisfied using a single subspace. We begin this section by giving sufficient conditions under which stagnation of the linear iteration never occurs.

The simplest condition is simply to ensure that the steepest descent direction $-g = -J^T F$ belongs to the subspace K. Then the minimum of $\|F + Jp\|_2$ for $p \in K$, is reduced from its value of $\|F\|_2$ when $p = 0$ to an amount not less than that obtained by a steepest descent step, i.e.,

$$\min_{p \in K} \|F + Jp\|_2 \leq \min_{\lambda \in \mathbb{R}} \|F + \lambda J J^T F\|_2 \leq \frac{\text{cond}_2(J)^2 - 1}{\text{cond}_2(J)^2 + 1} \|F\|_2. \tag{77}$$

As a result, if the steepest descent direction is known, or computable, it suffices to add it to the subspace to guarantee the existence of an $\eta \in (0, 1)$ for which the residual condition is always satisfied. The resulting modification of the underlying algorithms is very simple.

A particular case of the above situation is when $v_1 = \pm F/\|F\|_2$ and the Jacobian J is symmetric. Then the Krylov subspace K^m will in fact contain the steepest descent direction for the function $f = \frac{1}{2}F^T F$ for any $m \geq 2$. This is because

$$\nabla f = J^T F = JF = \pm \|F\|_2 J v_1$$

which, apart from a scaling factor, is the second vector of the Krylov sequence. As a result, stagnation can be avoided if J is symmetric and a Krylov subspace of dimension $m \geq 2$ is used. Analogous arguments also guarantee that the steepest descent direction lies in K whenever J is skew-symmetric or, whenever $J = I + \alpha S$, with S skew-symmetric, and α real. More generally, if there is a polynomial q of degree m such that $J^T = q(J)$ then again the steepest descent direction lies in K^m. However, this is equivalent to the property that J is normal; see [14] for details.

A third case for which there is no difficulty is when the Jacobian at every point is positive-real, i.e., $J + J^T$ is symmetric positive-definite. It has been shown by several authors that, in this situation, linear Krylov subspace methods will converge. Thus, if one required the Jacobian matrix $J(u)$ to be positive-real for all $u \in \mathbb{R}^N$, then the residual conditions in the main results of the previous sections can be satisfied, and so convergence of the sequence of nonlinear iterates will follow. This will be the case for some problems, but it is clear that requiring $J(u)$ to be positive real everywhere is very restrictive.

For problems whose coefficient matrices are indefinite, the convergence theory for the linear Arnoldi and GMRES algorithms is lacking in one very important area at the present time in that it is impossible to predict how well either algorithm will perform on the problem

$$Ax = b.$$

Some partial answers to this question have been given by several authors. For example, in [2], several results are shown which basically show that either both algorithms will perform well on a particular problem or both will perform poorly. Some numerical studies performed by Huang and Van Der Vorst [16] suggest that the convex hull of the upper Hessenberg matrix H_m in the Arnoldi process must well approximate that of the matrix A in order for the GMRES solution to well approximate the true solution of the above linear system.

There also are generalizations of the above results for the linear Krylov methods when the coefficient matrix is positive definite. We now show a result which gives necessary and sufficient conditions for the first k steps of the linear GMRES algorithm to stagnate.

THEOREM 5.1. *Consider the solution of $Ax = b$ using GMRES with initial guess x_0 and residual $r_0 = b - Ax_0$. Let x_i be the ith GMRES iterate for $i = 1, \ldots, k$. Assume A is nonsingular and let μ be the minimal degree of A with respect to r_0. Then for any $k \leq \mu$, we have $x_k = x_{k-1} = \cdots = x_0$ if and only if $r_0^T A^i r_0 = 0$ for $i = 1, \ldots, k$.*

Proof. Let $r_j = b - Ax_j$ $(j = 1, \ldots, k)$ with $k \leq \mu$. Because $j \leq k \leq \mu$, the $N \times j$ matrix $Z_j \equiv [r_0, Ar_0, \ldots, A^{k-1} r_0]$ is of full rank. It follows directly from the remark after (10) that $x_j = x_0 + Z_j y_j$, where $y_j \in \mathbb{R}^j$ solves the $j \times j$ system

$$(AZ_j)^T (r_0 - AZ_j y_j) = 0.$$

Since Z_j is of full rank and A is nonsingular, the above system has a unique solution. Observe that $x_j = x_0$ if and only if this unique solution is $y_j = 0$. This is true if

and only if the right-hand side $(AZ_j)^T r_0$ is zero, i.e., if and only if $r_0^T A^i r_0 = 0$ for $i = 1, \ldots, j$. The result follows immediately. □

Using this result, it follows immediately that if A^k is either positive or negative real, then GMRES(k') converges for any $k' \geq k$. Although interesting theoretically, this result is of limited practical application. However, it and the ones referred to above indicate some of the subtleties involved in analyzing the convergence behavior of the linear Krylov methods.

An important issue somewhat related to the question of stagnation is that of preconditioning. We mentioned above that adding the steepest descent direction to each subspace K guarantees convergence. In fact, using the normal equation approach, i.e., solving the linear systems by using the conjugate gradient algorithm on the normal equations, also guarantees convergence. However, the convergence can be very slow and this can be just as problematic as divergence. The usual way to improve convergence of the linear solvers is to precondition the systems. The key to avoiding stagnation is to use a good preconditioning technique. The importance of preconditioners over the Krylov subspace methods themselves has been illustrated in the tests in our previous paper [5]. There are a number of standard preconditioners that can be used when the Jacobian is explicitly available, the simplest and often the most inexpensive being the incomplete LU factorization. Unfortunately, when the Jacobian matrix is not available, i.e., in matrix-free methods, this is not feasible. Preconditionings that do not require the Jacobian explicitly but only its action on a given vector could be extremely useful. We should mention that any nonlinear fixed-point iteration can be considered as a matrix-free preconditioner [7]. We believe that much research remains to be done in this direction. For now, we can say that the best successes of nonlinear Krylov subspace methods have been in cases where the particular knowledge of the physics of the problem allows one to derive suitable preconditioners [32].

6. Conclusion. We have provided some theory for nonlinear projection methods with emphasis on those methods based on Krylov subspaces. The main results are similar to others in the literature, from which they have been adapted.

One of the main restrictions of most of the schemes used is that the subspace onto which a given Newton step is projected must solve the Newton equations with a certain accuracy which is dictated by the residual condition (4). This, as we have shown, is enough to essentially guarantee convergence of the standard linesearch and trust region algorithms. On the practical side, the main difficulty is that one does not know in advance if the subspace chosen for projection will be good enough to guarantee this residual condition. Techniques which use restarting of the linear iteration can be very useful in this context. Moreover, preconditioning is essential in the successful application of these methods.

Finally, there are generalizations of the Newton–Krylov methods considered in this paper which may be more effective on certain problems. For example, once a subspace K has been constructed along with an orthonormal basis V, one could consider solving the nonlinear least squares problem

$$\min_{y \in \mathbf{R}^m} f(u + Vy),$$

where $f(u) = \frac{1}{2} F(u)^T F(u)$ and u is the current approximate solution. This would be in lieu of the quadratic models considered in the trust region algorithms. Preliminary testing of such an algorithm has been encouraging. We will consider this and other generalizations in future work.

Appendix. The proof below follows closely that of Theorem 4.1 given in [29]. The major differences arise from the fact that a lower dimensional quadratic model is used, rather than the full N-dimensional model assumed in [29].

Proof of Theorem 4.4. By Taylor's theorem, for any n and any $\tau > 0$,

$$\begin{aligned}
&|\text{ared}_n(\tau) - \text{pred}_n(\tau)| \\
&\quad = \left| f_n - f(u_n + p_n(\tau)) - \left(f_n - f_n - g_n^T p_n(\tau) - \frac{1}{2} p_n(\tau)^T B_n p_n(\tau) \right) \right| \\
&\quad = \left| \frac{1}{2} p_n(\tau)^T B_n p_n(\tau) - \int_0^1 p_n(\tau)^T \nabla^2 f(u_n + \xi p_n(\tau)) p_n(\tau) \cdot (1-\xi) d\xi \right| \\
&\quad \le \|p_n(\tau)\|_2^2 \int_0^1 \|B_n - \nabla^2 f(u_n + \xi p_n(\tau))\|_2 (1-\xi) d\xi.
\end{aligned}$$

So

$$\left| \frac{\text{ared}_n(\tau)}{\text{pred}_n(\tau)} - 1 \right| \le \frac{\|p_n(\tau)\|_2^2 \int_0^1 \|B_n - \nabla^2 f(u_n + \xi p_n(\tau))\|_2 (1-\xi) d\xi}{|\text{pred}_n(\tau)|}. \tag{A.1}$$

Also, note that for any sequence $\{u_n\}$ generated by an algorithm satisfying the conditions of Algorithm 4.2, the related sequence $\{f_n\}$ is monotonically decreasing and bounded from below. Hence, f_n converges to an f_* as n goes to infinity. This fact will be used in the remainder of the proof.

Proof of (i). Since $\eta_n \le \eta_{\max} < 1$ for all n, we have

$$\sigma_n \ge \frac{1 - \eta_{\max}}{(1 + \eta_{\max}) M} \equiv \bar{\sigma} > 0 \quad \text{for all } n.$$

From (76) it follows that

$$f_n - \phi_n(q_n(\tau)) \ge \frac{1}{2} \bar{\sigma} \|g_n\|_2 \min \left\{ \tau, \bar{\sigma} \frac{\|g_n\|_2}{\|B_n\|_2} \right\} \quad \text{for all } n. \tag{A.2}$$

Next, consider any k with $\|g_k\|_2 \ne 0$. For any u, $\|g(u) - g_k\|_2 \le \beta_1 \|u - u_k\|_2$. So if $\|u - u_k\|_2 < \|g_k\|_2/(2\beta_1)$, then

$$\|g(u)\|_2 \ge \|g_k\|_2 - \|g(u) - g_k\|_2 \ge \frac{\|g_k\|_2}{2}.$$

Let $R = \|g_k\|_2/(2\beta_1)$ and $D_R = \{u : \|u - u_k\|_2 < R\}$.

At this point there are two possibilities: either $u_n \in D_R$ for all $n \ge k$ or eventually $\{u_n\}$ leaves the ball. We show the latter is true by contradiction. Suppose $u_n \in D_R$ for all $n \ge k$. Then for all $n \ge k$, $\|g_n\|_2 \ge \|g_k\|_2/2 (\equiv \bar{\epsilon})$. Thus, by what was shown above,

$$\begin{aligned}
\text{pred}_n(\tau) &\ge \frac{1}{2} \bar{\sigma} \|g_n\|_2 \min \left\{ \tau, \bar{\sigma} \frac{\|g_n\|_2}{\|B_n\|_2} \right\} \\
&\ge \bar{\sigma} \bar{\epsilon} \min \left\{ \tau, \bar{\sigma} \frac{\bar{\epsilon}}{\beta_2} \right\}
\end{aligned}$$

for all $n \ge k$, since $\|B_n\|_2 \le \beta_2$ and $\|g_n\|_2 \ge \bar{\epsilon}$ implies $\|g_n\|_2/\|B_n\|_2 \ge \bar{\epsilon}/\beta_2$. To simplify notation let $\delta = \bar{\sigma}\bar{\epsilon}$. Then, using the above inequality we have

$$\left| \frac{\text{ared}_n(\tau)}{\text{pred}_n(\tau)} - 1 \right| \le \frac{\|p_n(\tau)\|_2^2 \int_0^1 \|B_n - \nabla^2 f(u_n + \xi p_n(\tau))\|_2 (1-\xi) d\xi}{\delta \min\{\tau, \delta/\beta_2\}}$$

$$\leq \frac{\tau^2(\beta_1+\beta_2)}{\delta \min\{\tau, \delta/\beta_2\}}$$
$$\leq \frac{\tau(\beta_1+\beta_2)}{\delta}$$

for all $n \geq k$ and $\tau \leq \delta/\beta_2$. This gives for τ sufficiently small and $n \geq k$ that

$$\frac{\text{ared}_n(\tau)}{\text{pred}_n(\tau)} > \alpha_2.$$

In addition, we have

$$\|(V_n^T B_n V_n)^{-1} V_n^T g_n\|_2 \geq \frac{\|V_n^T g_n\|_2}{\|V_n^T B_n V_n\|_2} \geq \bar{\sigma} \frac{\|g_n\|_2}{\|B_n\|_2} \geq \frac{\delta}{\beta_2},$$

so that for τ_n sufficiently small none of the conditions allowing decrease of τ_n in 5(b) of Algorithm 4.2 above can hold. It follows that τ_n is bounded away from 0. But, since

$$f_n - f_{n+1} = \text{ared}_n(\tau_n) \geq \alpha_1 \text{pred}_n(\tau_n) \geq \alpha_1 \delta \min\left\{\tau_n, \frac{\delta}{\beta_2}\right\}, \tag{A.3}$$

and since f is bounded from below, we must have $\tau_n \to 0$, which is a contradiction. Therefore, $\{u_n\}$ must eventually be outside D_R for some $n > k$.

Let $l+1$ be the first index after k with u_{l+1} not in D_R. Then

$$\begin{aligned} f(u_k) - f(u_{l+1}) &= \sum_{n=k}^{l} f(u_n) - f(u_{n+1}) \\ &\geq \sum_{n=k}^{l} \alpha_1 \text{pred}_n(\tau_n) \\ &\geq \sum_{n=k}^{l} \alpha_1 \delta \min\left\{\tau_n, \frac{\delta}{\beta_2}\right\}. \end{aligned}$$

Now, if $\tau_n \leq \delta/\beta_2$ for $k \leq n \leq l$, we have that

$$f(u_k) - f(u_{l+1}) \geq \alpha_1 \delta \sum_{n=k}^{l} \tau_n \geq \alpha_1 \delta R.$$

Otherwise, we have that $f(u_k) - f(u_{l+1}) \geq \alpha_1 \delta^2/\beta_2$. (That is, there exists at least one n with $\tau_n > \delta/\beta_2$.) In either case,

$$\begin{aligned} f(u_k) - f(u_{l+1}) &\geq \alpha_1 \delta \min\left\{R, \frac{\delta}{\beta_2}\right\} \\ &= \alpha_1 \bar{\sigma} \frac{\|g_k\|_2}{2} \min\left\{\frac{\|g_k\|_2}{2\beta_1}, \frac{\bar{\sigma}\|g_k\|_2}{2\beta_2}\right\} \\ &\geq \|g_k\|_2^2 \alpha_1 \frac{\bar{\sigma}}{4} \min\left\{\frac{1}{\beta_1}, \frac{\bar{\sigma}}{\beta_2}\right\}. \end{aligned}$$

By assumption, f is bounded below, and by construction f_n is monotonically decreasing. These imply that $f_n \to f_*$. Then by the preceding inequality

$$\|g_k\|_2^2 \leq \left[\alpha_1 \frac{\bar{\sigma}}{4} \min\{\frac{1}{\beta_1}, \frac{\bar{\sigma}}{\beta_2}\}\right]^{-1} \qquad (f_k - f_*).$$

Since this last inequality holds for any k with $\|g_k\|_2 \neq 0$, we have $g_n \to 0$ as $n \to \infty$.

Proof of (ii). By assumption, u_* is a limit point of $\{u_n\}$. We show first that u_n converges to u_*. By (i), $g(u_*) = 0$. Since $J(u_*)$ is nonsingular and $0 = g(u_*) = J(u_*)^T F(u_*)$, it follows that $F(u_*) = 0$. We then have $\nabla^2 f(u_*) = J(u_*)^T J(u_*)$, and hence is positive definite. Since $\nabla^2 f$ is continuous, there exists a $\delta_1 > 0$ such that if $\|u - u_*\|_2 < \delta_1$, then $\nabla^2 f(u)$ is positive-definite, and if $u \neq u_*$ then $g(u) \neq 0$. Let $D_1 = \{u : \|u - u_*\|_2 < \delta_1\}$.

Since $g(u_*) = 0$, we can find $\delta_2 > 0$ with $\delta_2 < \delta_1/4$ and $\|B(u)^{-1}\|_2 \cdot \|g(u)\|_2 < \delta_1/2$ for all $u \in D_2 = \{u : \|u - u_*\|_2 < \delta_2\}$, defining $B(u) = J(u)^T J(u)$.

Let $\{u_{n_j}\}$ be a subsequence of $\{u_n\}$ converging to u_*, and find j_0 such that $f(u_{n_{j_0}}) < \inf\{f(u) : u \in D_1 - D_2\}$ and $u_{n_{j_0}} \in D_2$. Consider any u_l with $l \geq n_{j_0}$, $u_l \in D_2$. We claim that $u_{l+1} \in D_2$, which implies that the entire sequence beyond $u_{n_{j_0}}$ is in D_2. Suppose that u_{l+1} is not in D_2. Since $f_{l+1} < f_{n_{j_0}}$, u_{l+1} is not in D_1 either. So

$$\begin{aligned} \tau_l \geq \|u_{l+1} - u_l\|_2 &\geq \|u_{l+1} - u_*\|_2 - \|u_l - u_*\|_2 \geq \delta_1 - \frac{\delta_1}{4} = \frac{3}{4}\delta_1 \\ &> \frac{\delta_1}{2} \geq \|B_l^{-1}\|_2 \cdot \|g(u_l)\|_2 \\ &\geq \|(V_l^T B_l V_l)^{-1} V_l^T g_l\|_2, \end{aligned}$$

letting $B_l = B(u_l)$. But, since the inexact Newton step is within the trust region, we have

$$u_{l+1} - u_l = p_l = -(V_l^T B_l V_l)^{-1} V_l^T g(u_l),$$

which contradicts the strict inequality above. Hence, for all $n \geq n_{j_0}$, $u_n \in D_2$. Since $f(u_n)$ is strictly decreasing to zero, u_* is the unique zero of f in D_2, and $\nabla^2 f(u_*)$ is positive definite. Thus, we have that u_n converges to u_*.

Next, we show that the convergence rate is superlinear if $\eta_n \to 0$ as $n \to \infty$. This is done by showing that eventually $\|(V_n^T B_n V_n)^{-1} V_n^T g_n\|_2$ will always be less than τ_n, and hence the inexact Newton step will always be taken. Since $J(u_*)$ is nonsingular, it follows from the results in [9] that the convergence rate of u_n to u_* is superlinear.

To show that eventually the inexact Newton step is always shorter than the trust radius, we need a particular lower bound on $\text{pred}_n(\tau)$. By the assumptions of (ii), for all n large enough, $B_n = J(u_n)^T J(u_n)$ is positive definite. Hence, either the inexact Newton step is longer than the trust radius, or $p_n(\tau)$ is the inexact Newton step. In either case,

$$\|p_n(\tau)\|_2 \leq \|(V_n^T B_n V_n)^{-1} V_n^T g_n\|_2 \leq \|(V_n^T B_n V_n)^{-1}\|_2 \cdot \|g_n\|_2,$$

since $\|V_n^T g_n\|_2 \leq \|g_n\|_2$. It follows that $\|g_n\|_2 \geq \|p_n(\tau)\|_2 / \|B_n^{-1}\|_2$ since $\|B_n^{-1}\|_2 \geq \|(V_n^T B_n V_n)^{-1}\|_2$. By what was shown in the proof of (i), for all n large enough we have

$$\text{pred}_n(\tau) \geq \frac{1}{2}\bar{\sigma} \frac{\|p_n(\tau)\|_2}{\|B_n^{-1}\|_2} \min\left\{\tau, \bar{\sigma} \frac{\|g_n\|_2}{\|B_n\|_2}\right\}$$

$$\geq \frac{1}{2}\bar{\sigma}\frac{\|p_n(\tau)\|_2}{\|B_n^{-1}\|_2}\min\left\{\|p_n(\tau)\|_2, \bar{\sigma}\frac{\|p_n(\tau)\|_2}{\|B_n\|_2\|B_n^{-1}\|_2}\right\}$$
$$\geq \frac{1}{2}\frac{\bar{\sigma}^2}{M^2}\frac{\|p_n(\tau)\|_2^2}{\|B_n^{-1}\|_2}.$$

So, by the continuity of $\nabla^2 f$, for all n large enough,

$$\text{pred}_n(\tau) \geq \frac{\bar{\sigma}^2}{4M^2}\frac{\|p_n(\tau)\|_2^2}{\|\nabla^2 f(u_*)^{-1}\|_2}. \tag{A.4}$$

Since $B_n = J(u_n)^T J(u_n)$, we can write

$$\nabla^2 f(u_n) = B_n + \sum_{i=1}^{N} F_i(u_n)\nabla^2 F_i(u_n).$$

Hence,

$$B_n - \nabla^2 f(u_n + \xi p_n(\tau)) = \nabla^2 f(u_n) - \nabla^2 f(u_n + \xi p_n(\tau)) + \sum_{i=1}^{N} F_i(u_n)\nabla^2 F_i(u_n).$$

If we define $\rho_n \equiv \|\sum_{i=1}^{N} F_i(u_n)\nabla^2 F_i(u_n)\|_2$, then by the argument leading up to (A.1) and Lipschitz continuity,

$$|\text{ared}_n(\tau) - \text{pred}_n(\tau)| \leq \tfrac{1}{2}\|p_n(\tau)\|_2^2\left(L\|p_n(\tau)\|_2 + \rho_n\right).$$

Since $F(u_*) = 0$, given any $\epsilon > 0$, there exists an n_0 such that $\rho_n \leq \epsilon$ for all $n \geq n_0$. Thus, for any $\tau > 0$ and $n \geq n_0$ (choosing n_0 larger, if necessary, to also guarantee that (A.4) holds),

$$\left|\frac{\text{ared}_n(\tau)}{\text{pred}_n(\tau)} - 1\right| \leq \frac{2M^2\|\nabla^2 f(u_*)^{-1}\|_2}{\bar{\sigma}^2}(L\|p_n(\tau)\|_2 + \rho_n)$$
$$\leq \frac{2M^2\|\nabla^2 f(u_*)^{-1}\|_2}{\bar{\sigma}^2}(L\tau + \epsilon).$$

Next, choose ϵ so that

$$\epsilon < \frac{(1-\alpha_2)\bar{\sigma}^2}{2M^2\|\nabla^2 f(u_*)^{-1}\|_2}.$$

Then define $\tilde{\tau}$ by

$$\tilde{\tau} = \gamma_1\frac{(1-\alpha_2)\bar{\sigma}^2 - 2M^2\|\nabla^2 f(u_*)^{-1}\|_2\epsilon}{2M^2\|\nabla^2 f(u_*)^{-1}\|_2 L}.$$

It follows easily that for any $0 < \tau \leq \tilde{\tau}/\gamma_1$ and $n \geq n_0 + 1$

$$\frac{\text{ared}_n(\tau)}{\text{pred}_n(\tau)} \geq \alpha_2.$$

Thus, by step 5(b) of Algorithm 4.2, if $\tau_{n-1} < \tilde{\tau}$, then τ_n will be less than τ_{n-1} only if $\tau_n \geq \|(V_{n-1}^T B_{n-1} V_{n-1})^{-1} V_{n-1}^T g_{n-1}\|_2$. It follows from the superlinear convergence of the inexact-Newton method that for u_{n-1} close enough to u_* and n large enough,

$$\|(V_n^T B_n V_n)^{-1} V_n^T g_n\|_2 < \|(V_{n-1}^T B_{n-1} V_{n-1})^{-1} V_{n-1}^T g_{n-1}\|_2.$$

Now, if τ_n is bounded away from 0 for all large n, then we are done. Otherwise, if for an arbitrarily large n τ_n is reduced, i.e., $\tau_n < \tau_{n-1}$, then we have

$$\tau_n \geq \|(V_{n-1}^T B_{n-1} V_{n-1})^{-1} V_{n-1}^T g_{n-1}\|_2 > \|(V_n^T B_n V_n)^{-1} V_n^T g_n\|_2,$$

and so the full inexact Newton step is taken. Inductively, this occurs for all subsequence iterates and superlinear convergence follows. It follows immediately from the results of [9] that the convergence is quadratic if $\eta_n = O(\|F(u_n)\|_2)$. □

Acknowledgments. The authors are indebted to Homer Walker for the very valuable help he provided them by carefully reading the manuscript and pointing out a few errors. In particular, the modification of the first step in Algorithm 3.1 was spurred by his remarks. As was mentioned earlier, the result in Lemma 3.9 is due to Walker. Finally, the authors would also like to thank the referees for their detailed review of the manuscript and suggestions for improvement.

REFERENCES

[1] P. N. BROWN, *A local convergence theory for combined inexact-Newton/finite-difference projection methods,* SIAM J. Numer. Anal., 24 (1987), pp. 407–434.

[2] ———, *A theoretical comparison of the Arnoldi and GMRES algorithms,* SIAM J. Sci. Statist. Comput., 12 (1991), pp. 58–78.

[3] P. N. BROWN AND A. C. HINDMARSH, *Matrix-free methods for stiff systems of ODEs,* SIAM J. Numer. Anal., 24 (1987), pp. 610–638.

[4] ———, *Reduced storage methods in stiff ODE systems,* J. Appl. Math. Comp., 31 (1989), pp. 40–91.

[5] P. N. BROWN AND Y. SAAD, *Hybrid Krylov methods for nonlinear systems of equations,* SIAM J. Sci. Statist. Comput., 11 (1990), pp. 450–481.

[6] R. G. CARTER, *Numerical optimization in Hilbert spaces using inexact function and gradient evaluations,* SIAM J. Numer. Anal., 18 (1991), pp. 251–265.

[7] T. F. CHAN AND K. R. JACKSON, *The use of iterative linear equation solvers in codes for large systems of stiff IVPs for ODEs,* SIAM J. Sci. Statist. Comput., 7 (1986), pp. 378–417.

[8] I. L. CHERN AND W. L. MIRANKER, *Dichotomy and Conjugate Gradients in the Stiff Initial Value Problem,* Tech. Report 8032-34917, IBM, Yorktown Heights, NY, 1980.

[9] R. S. DEMBO, S. C. EISENSTAT, AND T. STEIHAUG, *Inexact Newton methods,* SIAM J. Numer. Anal., 19 (1982), pp. 400–408.

[10] J. E. DENNIS AND J. J. MORÉ, *A characterization of superlinear convergence and its application to quasi-Newton methods,* Math. Comp., 28 (1974), pp. 549–560.

[11] J. E. DENNIS AND R. B. SCHNABEL, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations,* Prentice Hall, Englewood Cliffs, NJ, 1983.

[12] S. C. EISENSTAT AND H. F. WALKER, *Private communication,* 1989.

[13] ———, *Globally convergent inexact Newton methods,* Tech. Report /91/51/, Utah State University, Logan, UT, 1991.

[14] V. FABER AND T. MANTEUFFEL, *Necessary and sufficient conditions for the existence of a conjugate gradient method,* SIAM J. Numer. Anal., 21 (1984), pp. 352–361.

[15] W. C. GEAR AND Y. SAAD, *Iterative solution of linear equations in ODE codes,* SIAM J. Sci. Statist. Comput., 4 (1983), pp. 583–601.

[16] Y. HUANG AND H. VAN DER VORST, *Some observations on the convergence behavior of GMRES,* Tech. Report 89-09, Delft University of Technology, Faculty Tech. Math., Delft, The Netherlands, 1989.

[17] T. KERKHOVEN AND Y. SAAD, *Acceleration techniques for decoupling algorithms in semiconductor simulation,* CSRD report 684, University of Illinois at Urbana-Champaign, 1987.

[18] J. J. MORÉ, *Recent developments in algorithms and software for trust region methods,* in Mathematical Programming: State of the Art, A. Bachem, M. Grötschel, and B. Korte, eds., pp. 258–287, Springer-Verlag, New York, 1983.

[19] H. MUKAI AND E. POLAK, *A second order method for unconstrained optimization,* J. Optim. Theory Appl., 26 (1978), pp. 1–20.

[20] S. G. NASH, *Newton-like minimization via the Lanczos method,* SIAM J. Numer. Anal., 21 (1984), pp. 770–788.

[21] ———, *Preconditioning of truncated-Newton methods,* SIAM J. Sci. Statist. Comput., 6 (1985), pp. 599–616.

[22] A. NAVARRA, *An application of GMRES to indefinite linear problems in meteorology,* in Practical Iterative Methods for Large Scale Computations, D. Boley, D. Truhlar, Y. Saad, R. Wyatt, and L. Collins, eds., North-Holland, Amsterdam, 1989, pp. 321–328.

[23] D. P. O'LEARY, *A discrete Newton algorithm for minimizing a function of many variables,* Math. Programming, 23 (1982), pp. 20–33.

[24] J. M. ORTEGA AND W. C. RHEINBOLDT, *Iterative Solution of Nonlinear Equations in Several Variables,* Academic Press, New York, 1970.

[25] M. J. D. POWELL, *A hybrid method for nonlinear equations,* Numerical Methods for Nonlinear Equations, P. Rabinowitz, ed., Gordon and Breach, New York, 1970.

[26] ———, *Convergence properties of a class of minimization algorithms,* in Nonlinear Programming 2, O. Mangasarian, R. Meyer, and S. Robinson, eds., Academic Press, New York, 1975, pp. 1–27.

[27] Y. SAAD, *Krylov subspace methods for solving unsymmetric linear systems,* Math. Comp., 37 (1981), pp. 105–126.

[28] Y. SAAD AND M. H. SCHULTZ, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems,* SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869.

[29] G. A. SHULTZ, R. B. SCHNABEL, AND R. H. BYRD, *A family of trust-region-based algorithms for unconstrained minimization with strong global convergence properties,* SIAM J. Numer. Anal., 22 (1985), pp. 47–67.

[30] T. STEIHAUG, *The conjugate gradient method and trust regions in large scale optimization,* SIAM J. Numer. Anal., 20 (1983), pp. 626–637.

[31] H. F. WALKER, *Lecture Notes,* CS 745, Computer Science Dept., Yale University, New Haven, CT, 1989.

[32] L. B. WIGTON, D. P. YU, AND N. J. YOUNG, *GMRES acceleration of computational fluid dynamics codes,* Proceedings of the 1985 AIAA Conference, Denver 1985, AIAA, Denver, CO, 1985.

SIAM J. OPTIMIZATION
Vol. 4, No. 2, pp. 331–339, May 1994
© 1994 Society for Industrial and Applied Mathematics
006

ON THE RESOLUTION OF LINEARLY CONSTRAINED CONVEX MINIMIZATION PROBLEMS*

ANA FRIEDLANDER†, JOSÉ MARIO MARTÍNEZ†, AND SANDRA A. SANTOS†

Abstract. The problem of minimizing a twice differentiable convex function f is considered, subject to $Ax = b$, $x \geq 0$, where $A \in \mathbb{R}^{M \times N}$, M, N are large and the feasible region is bounded. It is proven that this problem is equivalent to a "primal-dual" box-constrained problem with $2N + M$ variables. The equivalent problem involves neither penalization parameters nor ad hoc multiplier estimators. This problem is solved using an algorithm for bound constrained minimization that can deal with many variables. Numerical experiments are presented.

Key words. large-scale linearly constrained optimization, box-constrained problems, optimality conditions

AMS subject classifications. 90C06, 90C25, 90C30

1. Introduction.

In this paper we consider the problem

$$\begin{cases} \text{Minimize } f(x) \\ \text{such that } Ax = b \\ \qquad\qquad x \geq 0, \end{cases} \tag{1.1}$$

where $f \in C^2(\mathbb{R}^N)$ is convex, $A \in \mathbb{R}^{M \times N}$ and the set $\Omega \equiv \{x \in \mathbb{R}^N$ such that $Ax = b, x \geq 0\}$ is nonempty and bounded.

We are especially interested in the case where M and N are very large and the structure of A is such that even sparse factorizations (see Duff, Erisman, and Reid [6]) are not possible. In these cases, active set methods (see Fletcher [7]) cannot be applied to (1.1).

A very important particular case of (1.1) is when f is a convex quadratic function. In fact, if we are able to solve efficiently large quadratic problems of the form (1.1), then we have the essential ingredient for developing efficient sequential quadratic programming methods for general large-scale nonlinear programming. See Dembo and Tulowitzki [5], Nickel and Tolle [15], Yabe, Yamaki, and Takahashi [16], etc.

In the last few years it has been recognized by the optimization community that minimizing a function with single box constraints $l \leq x \leq u$ is a problem whose efficient resolution in the large-scale case leads to very efficient practical algorithms for general nonlinear programming. See Conn, Gould, and Toint [1]–[4]. As a result of this observation, many efficient algorithms were developed for general box-constrained minimization and for minimizing quadratics with box constraints. See Conn, Gould, and Toint [1], [2], Moré and Toraldo [13], [14], etc. So, since efficient algorithms and software for box-constrained minimization are available, it is natural to try to reduce more difficult problems to single box-constrained ones, even at the expense of increasing the number of variables.

In this paper, we prove that the problem (1.1) is equivalent to a box-constrained problem that results from considering the minimization of the 2-norm of the nonlinear system represented by the conditions of Karush–Kuhn–Tucker, including the complementarity slack conditions.

* Received by the editors November 17, 1992; accepted for publication (in revised form) January 25, 1993. This work was supported by FAPESP grants 90-3724-6 and 91-2441-3, CNPq, FINEP and FAEP-UNICAMP.

† Department of Applied Mathematics, IMECC-UNICAMP, University of Campinas, CP 6065, 13081 Campinas SP, Brazil.

The proof of the equivalence theorem is given in §2. In §3 we describe a trust region algorithm for box-constrained optimization, and we explain how this algorithm is used to solve the problem (1.1). In §4, we apply the new technique to the estimation of linearly constrained solutions of some linear systems of equations. Conclusions are stated in §5.

Notation. $||.||$ will denote the 2-norm of vectors and matrices throughout this paper.

2. The equivalence theorem. The Karush–Kuhn–Tucker optimality conditions for (1.1) are

$$\begin{cases} \nabla f(x) + A^T y - z = 0, \\ Ax = b, \\ x^T z = 0, \\ x \geq 0, z \geq 0. \end{cases} \tag{2.1}$$

Due to the convexity of f, (2.1) are necessary and sufficient conditions for x being a minimizer of (1.1). $y \in \mathbb{R}^M$ is the vector of dual variables, and $z \in \mathbb{R}^N$ represents the complementary slack dual variables.

Now, since f is continuous and the feasible set $\Omega \equiv \{x \in \mathbb{R}^N | Ax = b, x \geq 0\}$ is nonempty and bounded, the problem (1.1) admits a global minimizer x, associated to multipliers y and slack variables z. Obviously, global minimizers of (1.1) coincide with global minimizers of the following problem:

$$\begin{cases} \text{Minimize } F(x,y,z) \equiv (1/2)[||\nabla f(x) + A^T y - z||^2 \\ \qquad\qquad + ||Ax - b||^2 + (x^T z)^2] \\ \text{such that } x \geq 0, z \geq 0. \end{cases} \tag{2.2}$$

Moreover, by (2.1), the value of the objective function of (2.2) is zero at a global minimizer.

Now, (2.2) is a minimization problem with simple constraints. For solving this type of problem, even in very large-scale situations, some efficient algorithms exist. See Conn, Gould, and Toint [2], Friedlander, Martínez, and Santos [12]. However, the objective function of (2.2) is not convex, and typical algorithms for solving box-constrained problems are convergent only to stationary points, not necessarily global minimizers. In the following theorem we prove that, fortunately, in this case, in spite of the nonconvexity of the function, stationary points and global minimizers are the same.

THEOREM 2.1. *If $f \in C^2(\mathbb{R}^n)$ is convex and Ω is nonempty and bounded, then* (2.2) *has at least one stationary (Karush–Kuhn–Tucker) point and every stationary point of* (2.2) *is a global minimizer.*

Proof. The first part is trivial. Since Ω is bounded and f is continuous, a global minimizer of (1.1) exists. This minimizer must satisfy (2.1) and, hence, it must be a global minimizer of (2.2).

Assume now that (x, y, z) is a stationary point of (2.2).

Therefore, there exist $\gamma, \mu \in \mathbb{R}^N$ such that

$$A^T(Ax - b) + \nabla^2 f(x)[\nabla f(x) + A^T y - z] + (x^T z)z - \gamma = 0, \tag{2.3}$$

$$A[\nabla f(x) + A^T y - z] = 0, \tag{2.4}$$

$$-[\nabla f(x) + A^T y - z] + (x^T z)x - \mu = 0, \tag{2.5}$$

$$\gamma^T x = 0, \tag{2.6}$$

$$\mu^T z = 0, \tag{2.7}$$

$$x \geq 0, z \geq 0, \gamma \geq 0, \mu \geq 0. \tag{2.8}$$

By (2.4) and (2.5) we have that

$$x^T z x - \mu \in \mathcal{N}, \tag{2.9}$$

where $\mathcal{N}$ is the null-space of A.

Therefore, premultiplying (2.3) by $x^T z x - \mu$, and using (2.5), we obtain that

$$(x^T z x - \mu)^T \nabla^2 f(x)(x^T z x - \mu) + (x^T z x - \mu)^T (x^T z z - \gamma) = 0. \tag{2.10}$$

Since $\nabla^2 f(x)$ is positive semidefinite, (2.10) implies that

$$(x^T z x - \mu)^T (x^T z z - \gamma) \leq 0.$$

So, by (2.6) and (2.7), we have that

$$(x^T z)^3 + \mu^T \gamma \leq 0. \tag{2.11}$$

So, by (2.8),

$$x^T z = 0 \tag{2.12}$$

and

$$\mu^T \gamma = 0. \tag{2.13}$$

By (2.5) and (2.12),

$$-(\nabla f(x) + A^T y - z) = \mu \geq 0. \tag{2.14}$$

But, by (2.4), $-(\nabla f(x) + A^T y - z) \in \mathcal{N}$; therefore, since Ω is bounded, (2.14) necessarily implies that

$$-(\nabla f(x) + A^T y - z) = 0. \tag{2.15}$$

So, by (2.3), (2.12), and (2.15),

$$A^T(Ax - b) = \gamma \geq 0. \tag{2.16}$$

Now, (2.16) and (2.6) are the necessary and sufficient optimality conditions of the convex quadratic problem

$$\begin{cases} \text{Minimize } \frac{1}{2}||Ax - b||^2 \\ \text{such that } x \geq 0. \end{cases} \tag{2.17}$$

So, since Ω is not empty, we necessarily have that $Ax = b$. This equality, together with (2.12) and (2.15), completes the proof. □

Remark. The problem

$$\begin{cases} \text{Minimize } (\tfrac{1}{2})[||\nabla f(x) + A^T y - z||^2 + ||Ax - b||^2 + x^T z] \\ \text{such that } x \geq 0, z \geq 0 \end{cases} \tag{2.18}$$

is obviously equivalent to (2.2). However, it is very interesting to notice that (2.18) may have stationary points that are not global minimizers. As an example consider the problem of minimizing x, subject to $2 \geq x \geq 0$. The problem of the form (2.18) associated to this trivial problem admits the stationary point $x = 2, z = 0$, which, of course, does not represent a global minimizer of (2.18).

3. Algorithm for box-constrained minimization. In this section, we describe an algorithm for finding approximate solutions of the box-constrained minimization problem

$$\begin{cases} \text{Minimize } f(x) \\ \text{such that } l \leq x \leq u, \end{cases} \tag{3.1}$$

where some components of the vectors that define the bounds, l and u may be infinite. Clearly, with some abuse of notation (replace (x, y, z) by x), the problem (2.2) is of the form (3.1).

The algorithm that we are going to describe is of the trust region type and has been fully analyzed in Friedlander, Martínez, and Santos [12]. Generally speaking, it can be considered a variation of the trust region method for box-constrained problems of Conn, Gould, and Toint [1], [2], which is a part of the package LANCELOT. See Conn, Gould, and Toint [3], [4]. Here we intend only to give a brief description of the method, stating the theoretical properties that make it suitable for finding global solutions of (3.1).

Let $\sigma_1, \sigma_2, \alpha, \Delta_{\min}, \theta$ be such that $0 < \sigma_1 \leq \sigma_2 < 1, \alpha \in (0, 1), \Delta_{\min} > 0$, and $\theta \in (0, 1]$. At the beginning of the trust region algorithm we have an arbitrary initial feasible point x_0, an initial symmetric $n \times n$ matrix B_0 (the Hessian approximation), a nonsingular $n \times n$ matrix D_0 (the scaling matrix) and an initial radius $\Delta^\circ \geq \Delta_{\min}$. Given a feasible x_k, B_k symmetric, D_k nonsingular and $\Delta^k \geq \Delta_{\min}$, the steps for obtaining x_{k+1} and Δ_k are given by the following algorithm.

ALGORITHM 3.1.

Step 1. (Set the initial radius of the trust region and compute an upper bound of $||B_k||$.)

$$\text{Set } \Delta \leftarrow \Delta^k.$$

Compute $M_k > 0$ such that

$$||B_k|| \leq M_k. \tag{3.2}$$

Step 2. (Solve the "easy" subproblem.) Compute a global solution $z_k^Q(\Delta)$ of

$$\begin{cases} \text{Minimize } Q_k(z) \equiv \tfrac{1}{2} M_k ||z||^2 + g_k^T z \\ \text{such that } l \leq x_k + z \leq u, \\ \qquad\qquad ||D_k z||_\infty \leq \Delta, \end{cases} \tag{3.3}$$

where $g_k = g(x_k)$. If $Q_k(z_k^Q(\Delta)) = 0$, stop.

Step 3. (Compute the trial step.) Compute $\overline{z}_k(\Delta)$ such that

$$\begin{cases} \psi_k(\overline{z}_k(\Delta)) \leq \theta Q_k(z_k^Q(\Delta)), \\ l \leq x_k + \overline{z}_k(\Delta) \leq u, \\ ||D_k \overline{z}_k(\Delta)||_\infty \leq \Delta, \end{cases} \tag{3.4}$$

where

$$\psi_k(z) = \frac{1}{2} z^T B_k z + g_k^T z \tag{3.5}$$

for all $z \in \mathbb{R}^n$. (Observe that $z_k(\Delta)$ satisfying (2.3) exists, since $z_k^Q(\Delta)$ is an admissible choice.)

Step 4. (Test sufficient decrease.) If

$$f(x_k + \overline{z}_k(\Delta)) \leq f(x_k) + \alpha \psi_k(\overline{z}_k(\Delta)), \tag{3.6}$$

then define $z_k = \overline{z}_k(\Delta), x_{k+1} = x_k + z_k, \Delta_k = \Delta$, and return. Else, replace Δ by Δ_{new}, where

$$\Delta_{\text{new}} \in [\sigma_1 ||D_k \overline{z}_k(\Delta)||_\infty, \sigma_2 \Delta] \tag{3.7}$$

and repeat Step 2.

The following results are proved in Friedlander, Martínez, and Santos [12]. Let us recall that the main convergence Theorem 3.4 also holds for the LANCELOT box-constrained algorithm and, in fact, that it is the type of result that one should expect from every reasonable method for box-constrained minimization.

LEMMA 3.2. *If Algorithm* 3.1 *stops at Step* 2 (*so* $Q_k(z_k^Q(\Delta) = 0$), *then* x_k *is a Karush–Kuhn–Tucker point of the problem* (3.1).

THEOREM 3.3. *Algorithm* 3.1 *is well defined. That is, if the process does not stop at Step* 2 (*with* $Q_k(z_k^Q(\Delta)) = 0$) *then* x_{k+1} *can be computed repeating Steps* 2–4 *a finite number of times.*

THEOREM 3.4. *Assume that* $\{x_k\}$ *is an infinite sequence generated by Algorithm* 3.1. K_1 *is an infinite set of indices such that* $\lim_{k \in K_1} x_k = x^*$ *and* $M_k, ||D_k||_\infty, ||D_k^{-1}||_\infty$ *and* M_k *are bounded for* $k \in K_1$. *Then* x^* *is a stationary* (*Karush–Kuhn–Tucker*) *point of* (3.1).

A FORTRAN (double precision) code was written that implements Algorithm 3.1, where we used $\sigma_1 = \sigma_2 = 0.5, \alpha = 10^{-4}, \theta = 1, D_k = I, \Delta_{\min} = 0.1, \Delta^0 = 1, \Delta^{k+1} = \max\{\Delta_{\min}, 2\Delta_k\}$ for all $k = 0, 1, 2, \ldots$.

For computing $\overline{z}_k(\Delta)$ at Step 3 of Algorithm 3.1, we considered the following problem:

$$\begin{cases} \text{Minimize } \psi_k(z) \\ \text{such that } l \leq x_k + z \leq u, \\ \qquad\qquad ||D_k z||_\infty \leq \Delta. \end{cases} \tag{3.8}$$

For approximately solving (3.8) we used an algorithm that minimizes quadratics on a box. See Friedlander and Martínez [11], Friedlander, Martínez, and Santos [12]. This quadratic minimization solver proceeds by combining conjugate gradient iterations and projected gradients. So, only matrix × vector products are required,

no matrix factorizations are used and, in consequence, fill-in is not produced. The stopping criterion for the algorithm that solves (3.8) was

$$||P\nabla\psi_k(z)|| \leq 0.1||P\nabla\psi_k(0)||,$$

where P is the projection on the feasible set of (3.8). Of course, we also required the condition (3.4) of Algorithm 3.1.

In our experiments, the initial approximation z^0 for Algorithm 4.1 was $z_k^Q(\Delta)$. This choice is reasonable because it guarantees that the condition (3.4) required for convergence will be satisfied. Finally, at each call of the quadratic minimization solver, a maximum of n iterations was permitted. So, when n iterations are performed and the convergence criterion is not satisfied, we simply choose $\overline{z}_k(\Delta) = z^n$, since z^n obviously satisfies (3.4).

4. Numerical experiments. We are interested in the problem of finding robust estimators of parameters in problems that are modelled by large overdetermined linear systems of equations, subject to linear constraints and bounds on the variables.

Assume that the overdetermined linear system is given by

$$Hx = c, \tag{4.1}$$

and the constraints are

$$Ax = b, \tag{4.2}$$

$$x \geq 0, \tag{4.3}$$

where $A \in \mathbb{R}^{M\times N}$ and $H \in \mathbb{R}^{L\times N}$, and we assume that the set defined by (4.2) and (4.3) is nonempty and bounded. We used $M = N/2$ and $L = 2N$.

For estimating the parameters $x_i, i = 1,\ldots,N$ we could solve the following quadratic programming problem:

$$\begin{cases} \text{Minimize } \frac{1}{2}||Hx - c||^2 \\ \text{such that } Ax = b, \\ \qquad\qquad x \geq 0. \end{cases} \tag{4.4}$$

However, as is well known, the quadratic error objective function is very sensitive to extreme values on the observations ("outliers"), and so it is better to choose a function that does not have such a disadvantage.

This led us to consider the following problem:

$$\begin{cases} \text{Minimize } \phi([Hx - c]_1) + \cdots + \phi([Hx - c]_L) \\ \text{such that } Ax = b, \\ \qquad\qquad x \geq 0, \end{cases} \tag{4.5}$$

where $\phi(t) = \log(\cosh(t))$. See Green [8], [9], Lange [10]. The objective function is convex and twice differentiable. So, we can apply the techniques explained in §§2 and 3 for solving (4.5).

We generated a set of test problems as follows.

(a) The (i,j) entry of A is $i - j$. The (i,j) entry of H is $(i - 3j)/(i + 3j)$.

(b) A random "solution" x^* with entries between 0 and 10 is generated.

(c) b is calculated as the product Ax^*.

(d) We set $c^* = Hx^*$ and $c_i = c_i^*(1 + r_i PERT/100)$, where $PERT$ represents a percentual perturbation on the "true" vector c^* and r_i is random between -1 and 1.

Different problems were generated varying N and $PERT$. Each problem was transformed to the form (2.2) and solved using Algorithm 3.1. The Hessian approximations used in (3.5) were true analytic Hessians, except in that the terms that involved third-order derivatives of f were discarded. This simplification is completely justified from the local convergence point of view since every third-order term is multiplied by a factor that vanishes at the solution. So, it is expected that the behavior of the process should be very similar if we had used the more costly complete Hessians.

As initial approximation for the solution of (2.2) we set $x_0 = 0, y_0 = 0, z_0 = 0$. The iterative procedure was stopped when one of the following conditions was achieved.

(a) The norm of the projected gradient of F in (2.2) is less than

$$10^{-4} \max\{1, |F(x_k, y_k, z_k)|\} / \max\{1, ||(x_k, y_k, z_k)||\}.$$

(b) $F(x_k, y_k, z_k) \leq 10^{-8}$.

(c) $\Delta_k \leq 10^{-4}$.

TABLE 1
Numerical experiments.

N	*PERT*	ITER	FE	V1	V2	LF
6	0	18	27	4.1E-9	1.7E-5	1.0E-7
	10	22	30	1.4E-7	4.0E-1	8.2E-6
	20	14	20	1.2E-5	1.5	8.6E-5
	30	13	18	8.3E-6	2.9	6.2E-5
	40	15	21	1.9E-6	4.4	2.4E-5
	50	16	22	8.4E-6	6.1	5.0E-5
12	0	23	34	5.0E-12	5.0E-6	6.5E-10
	10	23	30	2.2E-4	6.4	4.5E-6
	20	41	52	1.2E-5	17.58	7.0E-6
	30	45	61	9.2E-6	30.75	2.2E-6
	40	42	57	9.7E-5	45.36	1.2E-5
	50	33	39	1.2E-9	59.96	7.0E-7
18	0	38	78	2.2E-11	3.5E-6	2.6E-7
	10	45	66	5.3E-7	21.81	1.9E-6
	20	41	72	1.3E-5	54.91	1.3E-5
	30	56	72	8.2E-6	90.85	8.8E-7
	40	39	70	5.1E-4	129.20	3.3E-5
	50	70	97	2.3E-5	167.03	1.0E-6
24	0	31	50	6.6E-9	9.6E-5	2.3E-8
	10	31	44	1.8E-4	45.19	1.4E-6
	20	53	73	1.7E-4	108.71	7.7E-7
	30	74	101	1.1E-4	174.40	1.1E-7
	40	58	87	1.2E-4	240.93	1.3E-6
	50	56	79	3.1E-5	307.77	1.4E-6
30	0	52	92	1.5E-7	8.5E-4	2.6E-7
	10	95	128	2.1E-4	83.87	1.9E-6
	20	55	103	1.8E-2	200.46	3.2E-5
	30	83	94	2.9E-4	312.12	2.8E-7
	40	113	181	3.6E-4	428.58	1.8E-5
	50	91	152	2.5E-4	545.57	1.7E-5

The results are given in Table 1. For each test we report N, $PERT$, and the following exit information:

(i) ITER: Number of iterations performed until convergence;

(ii) FE: Number of functional evaluations;

(iii) V1: Function value at the obtained point $(x, y, z)_{FINAL}$ for problem 2.2;
(iv) V2: Final function value obtained at the point x_{FINAL} for problem 4.5;
(v) LF: $||Ax_{FINAL} - b||$.

5. Final remarks. In this paper we showed that linearly constrained convex minimization problems can be solved without making use of matrix factorizations, provided that we have a reliable algorithm for minimizing arbitrary functions subject to box constraints, even if this algorithm only guarantees convergence to stationary points, as is usual. Since algorithms for solving large-scale box-constrained problems are becoming increasingly popular, we believe that our approach is interesting for solving problems when matrix factorizations are impossible. To the best of our knowledge, this is the first approach to this problem that does not need penalization parameters in any form.

Convex quadratic programming is just a particular case of the type of problems that we can solve, but many other convex problems may be relevant in applications.

Our numerical experiments seem to indicate that the proposed approach is reliable. Future research includes the application of the technique introduced here to the resolution of the subproblems in large-scale sequential quadratic programming methods.

Acknowledgments. We are indebted to Lúcio Tunes dos Santos and Álvaro De Pierro for helpful discussions.

REFERENCES

[1] A. R. CONN, N. I. M. GOULD, AND PH. L. TOINT, *Global convergence of a class of trust region algorithms for optimization with simple bounds*, SIAM J. Numer. Anal., 25 (1988), pp. 433-460; see also SIAM J. Numer. Anal., 26 (1989), pp. 764–767.

[2] ———, *Testing a class of methods for solving minimization problems with simple bounds on the variables*, Math. Comput., 50 (1989), pp. 399–430.

[3] ———, *A comprehensive description of* LANCELOT, Tech. Report 91/10, Department of Mathematics, Facultés Universitaires de Namur, Belgium, 1991.

[4] ———, *A globally convergent augmented Lagrangian algorithm for optimization with general constraints and simple bounds*, SIAM J. Numer. Anal., 28 (1991), pp. 545–572.

[5] R. S. DEMBO AND U. TULOWITZKI, *Sequential truncated quadratic programming methods*, in Numerical Optimization, P. T. Boggs, R. H. Byrd, and R. B. Schnabel, eds., Society for Industrial and Applied Mathematics, Philadelphia, PA, 1984, pp. 83–101.

[6] I. S. DUFF, A. M. ERISMAN, AND J. K. REID, *Direct Methods for Sparse Matrices*, Clarendon Press, Oxford, 1986.

[7] R. FLETCHER, *Practical Methods of Optimization*, 2nd ed., John Wiley and Sons, Chichester, New York, Brisbane, Toronto, and Singapore, 1987.

[8] P. GREEN, *On the use of the* EM *algorithm for penalized likelihood estimation*, J. Roy. Statist. Soc. B, 52 (1990), pp. 443–452.

[9] ———, *Bayesian reconstruction for emission tomography data using a modified* EM *algorithm*, IEEE Trans. Medical Imaging MI, 9 (1990), pp. 84–94.

[10] K. LANGE, *Convergence of* EM *image reconstruction algorithms with Gibbs smoothing*, IEEE Trans. Medical Imaging MI, 9 (1990), pp. 439–446.

[11] A. FRIEDLANDER AND J. M. MARTÍNEZ, *On the maximization of a concave quadratic with box constraints*, SIAM J. Optim., to appear.

[12] A. FRIEDLANDER, J. M. MARTÍNEZ, AND S. A. SANTOS, *A new trust region algorithm for bound constrained minimization*, Tech. Rep., Department of Applied Mathematics, University of Campinas, Brazil, 1992.

[13] J. J. MORÉ AND G. TORALDO, *Algorithms for bound constrained quadratic programming problems*, Numer. Math., 55 (1989), pp. 377–400.

[14] ———, *On the solution of large quadratic programming problems with bound constraints*, SIAM J. Optim., 1 (1991), pp. 93–113.

[15] R. H. NICKEL AND J. W. TOLLE, *A sparse sequential quadratic programming algorithm*, J. Optim. Theory Appl., 60 (1989), pp. 453–473.

[16] H. YABE, N. YAMAKI, AND S. TAKAHASHI, *Global convergence of sequential inexact* QP *method for constrained optimization*, SUT J. Math., 27 (1991), pp. 127–138.

SIAM J. OPTIMIZATION
Vol. 4, No. 2, pp. 340–357, May 1994
© 1994 Society for Industrial and Applied Mathematics
007

ON OPTIMIZATION PROBLEMS WITH VARIATIONAL INEQUALITY CONSTRAINTS*

JIŘÍ V. OUTRATA†

Abstract. This paper is devoted to a class of optimization problems that contain variational inequality or nonlinear complementarity constraints. Problems of this kind arise, for example, in game theory, bilevel programming, and the design of networks subject to equilibrium conditions. For such problems first-order necessary optimality conditions are derived and a numerical approach is proposed, based on nondifferentiable optimization techniques. This approach is illustrated by simple examples.

Key words. variational inequalities, sensitivity analysis, generalized Jacobian

AMS subject classifications. 49B34, 90C31, 93C30.

Introduction. In the foreword to the special issue of *Mathematical Programming* devoted to variational inequalities [5], as well as in the survey paper [11], the authors declare the mathematical programs with variational inequality/nonlinear complementarity constraints to be very difficult to solve due to their nonconvexity and nondifferentiability. However, such problems arise frequently, for example, in shape optimization, economic modelling, and design of transportation networks so that their study is of a definite practical interest.

As to our knowledge, a general problem of this type has been investigated from different points of view only in the papers [10], [29], [30], and [8], whereas various particular problems of this type are studied in [19], [20], [18], [31], [12], [7], [21], and [9]. Recently, we have learned of the collection of papers in [1], in which attention is also paid to this type of optimization problems.

In this paper a finite-dimensional optimization problem with a rather general variational inequality constraint is investigated. The aims follow:

(i) to derive necessary optimality conditions and

(ii) to propose an approach to the numerical solution of such problems based on the usage of nondifferentiable optimization (NDO) methods.

Concerning item (i), first-order optimality conditions have been already derived in [19], [20] for a special class of variational inequalities and in [29], [30] for a more general case. However, the conditions of [29], [30], using the derivatives of set-valued maps, are rather complicated and consequently difficult to verify. In this paper we apply a different approach, taking into account our finite-dimensional problem setting and the structure of the constraint set in the variational inequality (given by equalities and inequalities). It enables one to make use of the powerful methods of nonsmooth analysis to a larger extent than in the cited works. The derived conditions are especially useful in examining approximate solutions obtained by available numerical methods—at a good approximation they should be satisfied with a high accuracy. A similar approach has been applied in [24] but to a considerably narrower class of problems.

* Received by the editors January 27, 1992; accepted for publication (in revised form) February 4, 1993.

† Mathematical Institute, University of Bayreuth, D-95440 Bayreuth, Germany (jiri.outrata@uni-bayreuth.de). This research was supported in part by grants from the German Scientific Foundation and the Czechoslovak Academy of Sciences.

Concerning item (ii), for the numerical solution of the investigated problems various penalty techniques are available [10], [12] in which the used penalization depends on the nature of the variational inequality. Interesting heuristic algorithms have been developed in [18] and [8]. Our approach is closely connected with the derived optimality conditions, and in fact, it has already been successfully applied to the numerical solution of a certain class of Stackelberg problems [23], [15]. Nevertheless, the generalization performed opens a broad area of further important applications.

The study of variational inequalities itself represents an interesting topic in applied mathematics, closely connected with optimization. Fortunately, most of the basic questions have already been satisfactorily answered. From these important results we utilize in this paper especially those on sensitivity and stability, collected in [25] and [14]. We will also extensively refer to the survey [11].

The organization of the paper is as follows. In §1 we state the problem and impose the basic assumptions. Section 2 is devoted to the optimality conditions, and lastly the proposed numerical approach is explained in §3.

The following notation is employed: $\operatorname{conv}\Gamma$ denotes the convex hull of a set Γ, $f'(x_0; v)$ is the directional derivative of a function f at the point x_0 in the direction v, $L^{\perp}$ is the orthogonal complement to a linear subspace L, $\mathcal{N}(C)$ is the null space of a linear operator C, $\mathbb{R}^n_+$ is the nonnegative orthant of $\mathbb{R}^n$, A^i is the ith row of a matrix A, and x^i is the ith coordinate of a vector $x \in \mathbb{R}^n$. For $x, y \in \mathbb{R}^n$, $\langle x, y\rangle = x^T y$ and the inequalities $x \geq y$, $x > y$ mean $x^i \geq y^i$, $x^i > y^i$ for all i, respectively.

1. Problem formulation and basic assumptions. Assume that Γ is a nonempty convex polyhedral set in $\mathbb{R}^m$ and F is a continuously differentiable mapping from $A \times \mathbb{R}^m$ into $\mathbb{R}^m$, where A is an open subset of $\mathbb{R}^n$. Consider the parameter-dependent variational inequality:

For a given vector $\bar{x} \in A$, find a vector $\bar{y} \in \Gamma$ such that

$$\langle F(\bar{x}, \bar{y}),\ y - \bar{y}\rangle \geq 0 \quad \text{for all } y \in \Gamma. \tag{1.1}$$

Suppose that either Γ is compact or that for all $x \in A$, $F(x, \cdot)$ is *coercive with respect to* Γ, i.e., there exists a vector $y_x \in \Gamma$ such that

$$\lim_{\substack{y \in \Gamma \\ \|y\| \to \infty}} \langle F(x, y), y - y_x\rangle \,/\, \|y\| = \infty.$$

Then it has been proved in [22] that for all $x \in A$, problem (1.1) (with $\bar{x}$ replaced by x) possesses a nonempty, compact solution set. Thus, there is a proper, compact-valued map $S\,[A \rightsquigarrow \mathbb{R}^m]$ (cf. [3]), assigning to parameters $x \in A$ the (sets of) solutions y of the appropriate variational inequality (1.1). Consequently, one may define the optimization problem

$$\begin{aligned} &\text{minimize} && f(x, y) \\ &\text{subject to} && y \in S(x) \\ &&& x \in U_{\text{ad}}, \end{aligned} \tag{1.2}$$

where $f\,[\mathbb{R}^n \times \mathbb{R}^m \longrightarrow \mathbb{R}]$ is the objective function assumed to be continuously differentiable and $U_{\text{ad}} \subset A$ is a given nonempty compact set. The problem of the existence of solution pairs $(\hat{x}, \hat{y})$ for (1.2) is rather complicated and goes beyond the aims of this work. We confine ourselves therefore with a simple observation that (1.2) possesses a solution whenever S happens to be upper semicontinuous; cf. [3].

Instead of the form (1.1) we will prefer to write down our variational inequality in an equivalent form termed generalized equation [25]:

$$0 \in F(x,y) + N_\Gamma(y), \tag{1.3}$$

where for $y \in \Gamma$, $N_\Gamma(y)$ is the standard normal cone to Γ at y in the sense of convex analysis, and for $y \notin \Gamma$ we set $N_\Gamma(y) = \emptyset$. The reason is to facilitate the application of the strong results of Robinson [25], [26] regarding the behaviour of the map S.

Remark. The aforementioned results of Robinson can be directly used for the statement of necessary optimality conditions in terms of directional derivatives. The aim of the next section is, however, to derive a so-called dual form of these conditions, corresponding to the generalized Karush–Kuhn–Tucker conditions in [4].

At the end of this preliminary section we recall for the reader's convenience from [4] the definitions of the generalized gradient, the generalized Jacobian, and the tangent cone in the Clarke's sense and from [14] the concept of a nonsingular linear operator on a subspace.

DEFINITION 1.1. Let $H\,[\mathbb{R}^n \to \mathbb{R}^m]$ be a given operator, and let x_0 be a given point in $\mathbb{R}^n$. H is said to be *Lipschitz near* x_0 if there exist a scalar C and a neighbourhood $\mathcal{O}$ of x_0 such that

$$\|H(x'') - H(x')\| \leq C\|x'' - x'\| \quad \text{for all } x'', x' \in \mathcal{O}. \tag{1.4}$$

H is said to be *locally Lipschitz* on a subset Ω of $\mathbb{R}^n$, provided it is Lipschitz near each $x \in \Omega$.

DEFINITION 1.2. Let the operator $H[\mathbb{R}^n \to \mathbb{R}^m]$ be Lipschitz near $x_0 \in \mathbb{R}^n$, and let σ_H denote the set of points at which H fails to be Fréchet-differentiable. The *generalized Jacobian* of H at x_0, denoted $\partial H(x_0)$, is the set of $[m \times n]$ matrices, given by

$$\partial H(x_0) = \operatorname{conv}\left\{ \lim_{i\to\infty} \nabla H(x_i) \,|\, x_i \longrightarrow x_0,\ x_i \notin \sigma_H \right\}.$$

For $m = 1$, $\partial H(x_0)$ is termed the *generalized gradient* of H at x_0.

DEFINITION 1.3. Let $K \subset \mathbb{R}^n$ and $x \in K$. *Clarke's tangent cone* $T_K(x)$ is defined by

$$T_K(x) = \left\{ v \in \mathbb{R}^n \,|\, \lim_{\substack{\lambda \downarrow 0 \\ x' \to x, x' \in K}} \operatorname{dist}_K(x' + \lambda v)/\lambda = 0 \right\},$$

where $\operatorname{dist}_K(y) = \inf\{\|y - z\| \,|\, z \in K\}$.

Clarke's normal cone $N_K(x)$ is the negative polar cone to $T_K(x)$, i.e.,

$$N_K(x) = \{v^* \in \mathbb{R}^n \,|\, \langle v, v^* \rangle \leq 0 \text{ for all } v \in T_K(x)\}.$$

If K is convex, $N_K(x)$ coincides with the classical normal cone from convex analysis [4] so that we do not have any confusion with the notation introduced previously.

DEFINITION 1.4. Let $Q[\mathbb{R}^n \longrightarrow \mathbb{R}^n]$ be a linear operator, M be a linear subspace of $\mathbb{R}^n$, and Π_M denote the orthogonal projection operator onto M. We call Q *nonsingular on* M provided the map $\Pi_M Q$ is a bijection of M onto M.

In [14] we also have the following useful equivalent characterizations for Q to be nonsingular on M:

(C_1) for each $y \in \mathbb{R}^n$, there is a unique $z \in M$ such that $Qz - y \in M^\perp$;

(C_2) for each $z \neq 0$, $z \in M$, one has $Qz \notin M^\perp$.

2. Optimality conditions. Fix a pair (x_0, y_0) with $x_0 \in A$ and $y_0 \in S(x_0)$. According to [14] we introduce the cone

$$K(x_0, y_0) = T_\Gamma(y_0) \cap \{v \in \mathbb{R}^m \mid \langle F(x_0, y_0),\, v\rangle = 0\}$$

and the lineality space of $K(x_0, y_0)$

$$L(x_0, y_0) = K(x_0, y_0) \cap (-K(x_0, y_0))\ .$$

The following condition is equivalent to Robinson's original Strong Regularity Condition from [25] for polyhedral Γ; cf. [11], [14], and [26].

DEFINITION 2.1. The *Strong Regularity Condition* (SRC) holds at (x_0, y_0) for the generalized equation (1.3) if the linear generalized equation

$$u \in \nabla_y F(x_0, y_0)\, v + N_{K(x_0,y_0)}(v) \tag{2.1}$$

has a unique solution $v = v(u)$ for each $u \in \mathbb{R}^m$.

Assume now, additionally to the assumptions of the preceding section, that (SRC) holds at (x_0, y_0). Then, according to [25] and [14], there exist neighbourhoods $\mathcal{O}$ of x_0 and $\mathcal{M}$ of y_0 such that

(i) the map $\tilde{S}[\mathcal{O} \rightsquigarrow \mathcal{M}]$, defined by $\tilde{S}(x) = S(x) \cap \mathcal{M}$ for $x \in \mathcal{O}$, is Lipschitz and single valued;

(ii) and $\tilde{S}$ is directionally differentiable[1] at x_0 with the directional derivative $\tilde{S}'(x_0; h) = v$, which uniquely solves the linear generalized equation

$$0 \in \nabla_x F(x_0, y_0)\, h + \nabla_y F(x_0, y_0)\, v + N_{K(x_0,y_0)}(v). \tag{2.2}$$

Moreover, if (SRC) holds at (x_0, y_0), then the operator $\nabla_y F(x_0, y_0)$ is nonsingular on $L(x_0, y_0)$. Evidently, the same is true also for the adjoint operator $(\nabla_y F(x_0, y_0))^T$.

Thus, when deriving the necessary conditions for a local minimum (x_0, y_0) of (1.2), we may restrict ourselves to the problem

$$\begin{aligned} &\text{minimize} && \Theta(x) := f(x, \tilde{S}(x)) \\ &\text{subject to} && x \in U_{\text{ad}} \cap \mathcal{O}. \end{aligned} \tag{2.3}$$

As $\tilde{S}$ is generally nondifferentiable, the new objective Θ need not be differentiable as well. However, it is Lipschitz near x_0, and according to the Jacobian chain rule one has

$$\partial\Theta(x_0) = \nabla_x f(x_0, \tilde{S}(x_0)) + \left\{(P(x_0))^T \nabla_y f(x_0, \tilde{S}(x_0)) \mid P(x_0) \in \partial\tilde{S}(x_0)\right\} \tag{2.4}$$

(it suffices to express Θ as $f \circ G$, where $G : x \mapsto (x, \tilde{S}(x))$, and apply Theorem 2.6.6 from [4], taking the continuous differentiability of f into account). The derivation of first-order necessary optimality conditions thus amounts (with respect to the results of [13]) to evaluating a suitable upper approximation of the generalized Jacobian $\partial\tilde{S}(x_0)$. To this purpose we exploit another important result of [14, Thm. 2.2].

PROPOSITION 2.2. *Let all assumptions of* §1 *be satisfied, and let* (SRC) *hold at* (x_0, y_0). *Then the following statements are equivalent for the operator* $\tilde{S}$:

(i) $\tilde{S}$ *is Fréchet-differentiable at* x_0;

[1] In fact, $\tilde{S}$ is even B-differentiable (Bouligand-differentiable); cf. [14] and [26].

(ii) *for all directions* $h \in \mathbb{R}^n$, $\tilde{S}'(x_0; h) = v$ *is the unique solution of the linear generalized equation*

$$0 \in \nabla_x F(x_0, y_0)\, h + \nabla_y F(x_0, y_0)\, v + N_{L(x_0,y_0)}(v). \tag{2.5}$$

Thus, under the assumptions being imposed, the derivative $\nabla \tilde{S}(x_0)$ is the linear operator assigning to $h \in \mathbb{R}^n$ the unique solution v of (2.5) whenever $\tilde{S}$ is differentiable at x_0. Using this result, we can directly state the optimality conditions in the simple case, when $\tilde{S}$ is differentiable at the optimal vector $\hat{x}$.

THEOREM 2.3. *Let* $(\hat{x}, \hat{y})$ *be a (locally) optimal solution of problem* (1.2). *Suppose that all assumptions of* §1 *are satisfied,* (SRC) *holds at* $(\hat{x}, \hat{y})$, *and* $\tilde{S}$ *is differentiable at* $\hat{x}$. *Finally assume that* $p = \hat{p}$ *is the (unique) solution of the adjoint linear generalized equation*

$$0 \in (\nabla_y F(\hat{x}, \hat{y}))^T p + \nabla_y f(\hat{x}, \hat{y}) + N_{L(\hat{x},\hat{y})}(p). \tag{2.6}$$

Then one has

$$0 \in \nabla_x f(\hat{x}, \hat{y}) + (\nabla_x F(\hat{x}, \hat{y}))^T \hat{p} + N_{U_{\mathrm{ad}}}(\hat{x}). \tag{2.7}$$

Proof. We show that

$$\left(\nabla \tilde{S}(\hat{x})\right)^T \nabla_y f(\hat{x}, \hat{y}) = (\nabla_x F(\hat{x}, \hat{y}))^T \hat{p}.$$

Relation (2.7) then directly follows from (2.4) since for the optimal $\hat{x}$ one has $0 \in \partial\Theta(\hat{x}) + N_{U_{\mathrm{ad}}}(\hat{x})$; cf. [13].

We start with the observation that due to the nonsingularity of $(\nabla_y F(\hat{x}, \hat{y}))^T$ on $L(\hat{x}, \hat{y})$, the adjoint generalized equation (2.6) possesses a unique solution $\hat{p}$. Indeed, for any $p \in L(\hat{x}, \hat{y})$ one has $N_{L(\hat{x},\hat{y})}(p) = (L(\hat{x}, \hat{y}))^{\perp}$ so that (2.6) can be rewritten in the form

$$(\nabla_y F(\hat{x}, \hat{y}))^T p + \nabla_y f(\hat{x}, \hat{y}) \in (L(\hat{x}, \hat{y}))^{\perp}, \qquad p \in L(\hat{x}, \hat{y}).$$

However, by condition (C_1) stated in §1, there is a unique $\hat{p}$ satisfying these relations.

Now, by use of this adjoint generalized equation, for any $h \in \mathbb{R}^n$ one has

$$\left\langle \nabla_y f(\hat{x}, \hat{y}),\, \nabla \tilde{S}(\hat{x})\, h \right\rangle = \left\langle -(\nabla_y F(\hat{x}, \hat{y}))^T \hat{p} + \xi,\ \nabla \tilde{S}(\hat{x})\, h \right\rangle,$$

where $\xi \in (L(\hat{x}, \hat{y}))^{\perp}$. By Proposition 2.2 $\nabla \tilde{S}(\hat{x})h$ is the unique solution to the generalized equation (2.5), which implies that $\nabla \tilde{S}(\hat{x})h \in L(\hat{x}, \hat{y})$. Hence, $\langle \xi, \nabla \tilde{S}(\hat{x})h \rangle = 0$ and one has

$$\left\langle \nabla_y f(\hat{x}, \hat{y}), \nabla \tilde{S}(\hat{x})h \right\rangle = \left\langle \hat{p}, -(\nabla_y F(\hat{x}, \hat{y}))\, \nabla \tilde{S}(\hat{x})\, h \right\rangle.$$

Moreover, again due to (2.5),

$$-\nabla_y F(\hat{x}, \hat{y})\, \nabla \tilde{S}(\hat{x})\, h = \nabla_x F(\hat{x}, \hat{y})\, h + \eta$$

for some $\eta \in (L(\hat{x}, \hat{y}))^{\perp}$. Consequently

$$\left\langle \nabla_y f(\hat{x}, \hat{y}),\, \nabla \tilde{S}(\hat{x})\, h \right\rangle = \left\langle (\nabla_x F(\hat{x}, \hat{y}))^T \hat{p},\, h \right\rangle + \langle \hat{p}, \eta \rangle \quad \text{for all } h \in \mathbb{R}^n,$$

and we are done because $\hat{p} \in L(\hat{x}, \hat{y})$. □

To be able to deal with the nondifferentiable case, we provide Γ with the structure

$$\Gamma = \{y \in \mathbb{R}^m | A_1 y = b_1, A_2 y \leq b_2\}, \tag{2.8}$$

where matrices $A_1[p \times m]$, $A_2[s \times m]$ and vectors $b_1 \in \mathbb{R}^p$, $b_2 \in \mathbb{R}^s$ are given. Fortunately, this restriction does not affect the generality of the obtained results too much, because even variational inequalities with nonpolyhedral sets Γ can be most easily transformed to the required form. Indeed, let, for example,

$$\Gamma = \{y \in \mathbb{R}^m | G_1(y) = 0,\ G_2(y) \leq 0\},$$

where the functions $G_1[\mathbb{R}^m \to \mathbb{R}^p]$, $G_2[\mathbb{R}^m \to \mathbb{R}^s]$ are continuously differentiable. Then, under a suitable constraint qualification, the generalized equation (1.3) in variables (x, y) may be replaced by the generalized equation

$$0 \in \begin{bmatrix} F(x,y) + (\nabla G_1(y))^T \mu + (\nabla G_2(y))^T \lambda \\ G_1(y) \\ -G_2(y) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ N_{\mathbb{R}^s_+}(\lambda) \end{bmatrix}$$

in variables (x, y, μ, λ), which meets the above structural requirement.

We return now to Γ given by (2.8) and impose the following constraint qualification:

(LI) the rows A_1^i, $i = 1, 2, \ldots, p$, and A_2^i for $i \in I(x,y) := \{j \in \{1, 2, \ldots, s\} \mid \langle A_2^j, y\rangle = b_2^j\}$ are linearly independent at any $y \in S(x)$, $x \in A$.

Then, evidently, for $y \in S(x)$ one has

$$N_\Gamma(y) = \left\{v \in \mathbb{R}^m \mid v = A_1^T \mu + A_2^T \lambda, \text{ where } \mu \in \mathbb{R}^p \text{ and } \lambda \in \mathbb{R}^s_+ \text{ such that } \lambda^i = 0 \text{ for } i \notin I(x,y)\right\},$$

and with any $x_0 \in A$, $y_0 \in S(x_0)$ we can associate two unique multiplier vectors $\mu_0 \in \mathbb{R}^p$ and $\lambda_0 \in \mathbb{R}^s_+$ satisfying the Karush–Kuhn–Tucker (K.K.T.)-type conditions for (1.1)

$$\begin{aligned} 0 &= F(x_0, y_0) + A_1^T \mu_0 + A_2^T \lambda_0 \\ \lambda_0^i \left(\langle A_2^i, y_0\rangle - b_2^i\right) &= 0, \qquad i = 1, 2, \ldots, s. \end{aligned} \tag{2.9}$$

For $x \in A$, $y \in S(x)$ we introduce the index sets

$$J(x,y) := \left\{i \in I(x,y) \mid \lambda^i > 0\right\}, \tag{2.10}$$

where $\lambda \in \mathbb{R}^s_+$ is the (unique) multiplier vector determined by conditions (2.9) with (x_0, y_0) being replaced by (x, y). Sets (2.10) specify so-called *strongly active* inequality constraints, whereas the inequalities for $i \in I(x,y) \setminus J(x,y)$ are sometimes termed *semiactive*.

LEMMA 2.4. *Under the assumptions of Proposition* 2.2 *there exists a neighbourhood* $\mathcal{U}$ *of* x_0 *such that for all* $x \in \mathcal{U}$, $y = \tilde{S}(x)$ *one has*

$$J(x_0, y_0) \subset J(x,y) \subset I(x,y) \subset I(x_0, y_0). \tag{2.11}$$

Proof. Let $\mathcal{O}$ and $\mathcal{M}$ be the neighbourhoods for the operator $\tilde{S}$ relating to the (SRC) condition. Then the validity of the inclusion $I(x, \tilde{S}(x)) \subset I(x_0, y_0)$ on a neighbourhood $\tilde{\mathcal{U}}$ of x_0, $\tilde{\mathcal{U}} \subset \mathcal{O}$, is clearly implied by the continuity of $\tilde{S}$; thus it remains only to prove the inclusion on the left-hand side of (2.11).

To this purpose we show that the multiplier vector $\lambda \in \mathbb{R}^s_+$ assigned to the inequality constraint $A_2 y \leq b_2$ depends in a continuous way on x for $x \in \tilde{\mathcal{U}}$. Without any loss of generality, we omit the equality constraint $A_1 y = b_1$ in these considerations. We observe first that due to assumption (LI) there exists a constant $\beta > 0$ such that $\| \sum_{i \in I(x_0, y_0)} \kappa^i A_2^i \| \geq \beta \|\kappa\|$ for each κ of appropriate dimension. Therefore, from (2.9) for all $x \in \tilde{\mathcal{U}}$ one has

$$\left\| F(x, \tilde{S}(x)) \right\| \geq \beta \|\Lambda(x)\|, \tag{2.12}$$

where Λ assigns to x the unique multiplier vector $\lambda \in \mathbb{R}^s_+$ satisfying the K. K. T.-type conditions

$$\begin{aligned} 0 &= F(x, \tilde{S}(x)) + A_2^T \lambda \\ \lambda^i \left(\langle A_2^i, \tilde{S}(x) \rangle - b_2^i \right) &= 0, \qquad i = 1, 2, \ldots, s. \end{aligned}$$

As $\tilde{S}$ is continuous on $\tilde{\mathcal{U}}$ and F is continuous, (2.12) implies the existence of a constant $c > 0$ such that

$$\|\Lambda(x)\| \leq c \quad \text{for all } x \in \tilde{\mathcal{U}}. \tag{2.13}$$

Now consider a sequence $\{x_j\} \subset \tilde{\mathcal{U}}$, $x_j \to x_0$. Due to (2.13) we can select from $\{\Lambda(x_j)\}$ a subsequence $\{\Lambda(x_{j'})\}$, converging to a multiplier vector $\bar{\lambda} \in \mathbb{R}^s_+$. Evidently, for all j' one has

$$\begin{aligned} 0 &= F\left(x_{j'}, \tilde{S}(x_{j'})\right) + A_2^T \Lambda(x_{j'}) \\ \Lambda^i(x_{j'}) \left(\left\langle A_2^i, \tilde{S}(x_{j'}) \right\rangle - b_2^i \right) &= 0, \qquad i = 1, 2, \ldots, s, \end{aligned}$$

so that necessarily $\bar{\lambda}$ is a multiplier vector corresponding to the pair (x_0, y_0). Due to the (LI) condition this multiplier vector is unique, and thus for the whole sequence $\{x_j\}$ we obtain $\Lambda(x_j) \longrightarrow \Lambda(x_0)$.

The continuity of Λ on $\tilde{\mathcal{U}}$ implies the existence of a neighbourhood $\mathcal{U}$ of x_0, $\mathcal{U} \subset \tilde{\mathcal{U}}$, such that for $x \in \mathcal{U}$ the multipliers $\Lambda^i(x)$, $i \in J(x_0, y_0)$, are positive. Hence, for $x \in \mathcal{U}$ one has $J(x_0, y_0) \subset J(x, y)$, and the assertion has been proved. □

(SRC) plays an essential role in our approach to the optimality conditions. According to [11, Proposition 4.1] (where (SRC) is called just regularity), its validity at (x_0, y_0) may be ensured by the following hypothesis.

(H) For all $z \in \mathbb{R}^m \setminus \{0\}$ such that $A_1 z = 0$ and $\langle A_2^i, z \rangle = 0$ for $i \in J(x_0, y_0)$, one has

$$\langle z, \nabla_y F(x_0, y_0)\, z \rangle > 0.$$

Additionally, (H) has also other important consequences.

LEMMA 2.5. *Let all assumptions of* §1 *be satisfied, and at the pair* (x_0, y_0), $x_0 \in A$, $y_0 \in S(x_0)$, *let hypothesis* (H) *hold. Assume that the index set* $\tilde{J}$ *satisfies the*

inclusion $J(x_0, y_0) \subset \tilde{J} \subset I(x_0, y_0)$. *Then the operator* $\nabla_y F(x_0, y_0)$ *is nonsingular on the linear subspace*

$$L_{\tilde{J}}(x_0, y_0) = \left\{ z \in \mathbb{R}^m \mid A_1 z = 0,\ \langle A_2^i, z \rangle = 0 \text{ for } i \in \tilde{J} \right\}.$$

Proof. With respect to the characterization (C_2) of nonsingular operators in §1 it suffices to show that for each $z \neq 0$, $z \in L_{\tilde{J}}(x_0, y_0)$, one has $\nabla_y F(x_0, y_0)\, z \notin (L_{\tilde{J}}(x_0, y_0))^{\perp}$. Assume, by contradiction, the existence of a nonzero $\bar{z} \in L_{\tilde{J}}(x_0, y_0)$ such that

$$\nabla_y F(x_0, y_0)\, \bar{z} \in \operatorname{span} \left\{ A_1^{\ell},\ A_2^j \mid \ell = 1, 2, \ldots, p,\ j \in \tilde{J} \right\}.$$

Then there exist a vector $\alpha \in \mathbb{R}^p$ and scalars β^j, $j \in \tilde{J}$, such that

$$0 = \nabla_y F(x_0, y_0)\, \bar{z} + A_1^T \alpha + \sum_{i \in \tilde{J}} \beta^j A_2^j.$$

Multiplying each side of this equation by $\bar{z}^T$ we obtain

$$\langle \bar{z},\ \nabla_y F(x_0, y_0)\, \bar{z} \rangle = 0,$$

which contradicts (H). Thus, the assertion has been proved. □

COROLLARY 2.6. *Under the assumptions of Lemma* 2.5 *the linear generalized equations*

$$0 \in \nabla_y F(x_0, y_0)\, v + w + N_{L_{\tilde{J}}(x_0, y_0)}(v),$$
$$0 \in (\nabla_y F(x_0, y_0))^T\, p + w + N_{L_{\tilde{J}}(x_0, y_0)}(p)$$

possess unique solutions for each vector $w \in \mathbb{R}^m$.

Proof. It suffices to combine the assertion of Lemma 2.5 with characterization (C_1) of the nonsingular operators in §1. □

With the help of the assertions of Theorem 2.3 and Lemmas 2.4 and 2.5 we are now prepared to state the necessary optimality conditions for (1.2) in the general (nondifferentiable) case. Then, however, we have to consider the whole family $\mathcal{P}(I(\hat{x}, \hat{y}) \setminus J(\hat{x}, \hat{y}))$ of all subsets of $I(\hat{x}, \hat{y}) \setminus J(\hat{x}, \hat{y})$ at the optimal solution $(\hat{x}, \hat{y})$, because each subset of this family may generate a matrix from the generalized Jacobian of $\tilde{S}$ at $\hat{x}$. Therefore, we denote the single sets from $\mathcal{P}(I(\hat{x}, \hat{y}) \setminus J(\hat{x}, \hat{y}))$ by $T_i(\hat{x})$, where i runs through a suitably chosen index set $\mathbb{K}(\hat{x})$.

THEOREM 2.7. *Let* $(\hat{x}, \hat{y})$ *be a (locally) optimal solution of problem* (1.2) *with* Γ *being given by* (2.8). *Suppose that all assumptions of* §1 *are satisfied and* (H) *holds at* $(\hat{x}, \hat{y})$. *Furthermore, assume that vectors* $\hat{p}_i$, $i \in \mathbb{K}(\hat{x})$, *solve the adjoint generalized equations*

$$0 \in (\nabla_y F(\hat{x}, \hat{y}))^T\, p_i + \nabla_y f(\hat{x}, \hat{y}) + N_{L_i(\hat{x}, \hat{y})}(p_i), \tag{2.14}$$

where

$$L_i(\hat{x}, \hat{y}) = \left\{ z \in \mathbb{R}^m \mid A_1 z = 0,\ \langle A_2^j, z \rangle = 0,\ j \in T_i(\hat{x}) \cup J(\hat{x}, \hat{y}) \right\}.$$

Then one has

$$0 \in \nabla_x f(\hat{x}, \hat{y}) + (\nabla_x F(\hat{x}, \hat{y}))^T \operatorname{conv} \{ \hat{p}_i \mid i \in \mathbb{K}(\hat{x}) \} + N_{U_{\mathrm{ad}}}(\hat{x}).$$

Proof. Similarly as in Theorem 2.3 it suffices to show that

$$\left\{(P(\hat{x})^T \nabla_y f(\hat{x},\hat{y}) \,|\, P(\hat{x}) \in \partial\tilde{S}(\hat{x})\right\} \subset (\nabla_x F(\hat{x},\hat{y}))^T \operatorname{conv}\{\hat{p}_i \,|\, i \in \mathbb{K}(\hat{x})\}\,.$$

Let $\mathcal{U}$ be a neighbourhood of $\hat{x}$ specified by Lemma 2.4, and for $x \in \mathcal{U}$ let $y = \tilde{S}(x)$. Due to the assertion of Lemma 2.4 and the continuous differentiability of F it is clear that there exists a neighbourhood $\mathcal{V}$ of $\hat{x}$, $\mathcal{V} \subset \mathcal{U}$, where hypothesis (H) holds and thus also (SRC) is satisfied. The assertions of Proposition 2.2 and Lemma 2.4 now imply that at any $x \in \mathcal{V}$ at which $\tilde{S}$ is differentiable, there exists an index $i \in \mathbb{K}(\hat{x})$ such that $\nabla\tilde{S}(x)$ assigns to $h \in \mathbb{R}^n$ the unique solution v of the linear generalized equation

$$0 \in \nabla_x F(x,y)\, h + \nabla_y F(x,y)\, v + N_{L_i(\hat{x},\hat{y})}(v). \tag{2.15}$$

Consider now a sequence $\{x_j\} \subset \mathcal{V}$, $x_j \to \hat{x}$, such that $\tilde{S}$ is differentiable at x_j and $\nabla\tilde{S}(x_j)$ is specified by the generalized equation (2.15) for all x_j by the same index $i \in \mathbb{K}(\hat{x})$. In the next step of the proof we show that whenever $\nabla\tilde{S}(x_j)$ converges, the limit operator $P_i(\hat{x})$ satisfies for each $h \in \mathbb{R}^n$ the generalized equation

$$0 \in [\nabla_x F(\hat{x},\hat{y}) + \nabla_y F(\hat{x},\hat{y})\, P_i(\hat{x})]\; h + N_{L_i(\hat{x},\hat{y})}\,(P_i(\hat{x})h)\,. \tag{2.16}$$

Indeed, by denoting $\Omega(x_j) = \nabla_x F(x_j, \tilde{S}(x_j)) + \nabla_y F(x_j, \tilde{S}(x_j))\, \nabla\tilde{S}(x_j)$, Proposition 2.2 implies that for each $h \in \mathbb{R}^n$

$$\Omega(x_j)h \in (L_i(\hat{x},\hat{y}))^{\perp}\,, \qquad \nabla\tilde{S}(x_j)h \in L_i(\hat{x},\hat{y}).$$

As $\nabla\tilde{S}(x_j) \to P_i(\hat{x})$, the continuous differentiability of F and the continuity of $\tilde{S}$ near $\hat{x}$ yield

$$\Omega(x_j) \longrightarrow \nabla_x F(\hat{x},\hat{y}) + \nabla_y F(\hat{x},\hat{y})\, P_i(\hat{x}).$$

Due to the closedness of $L_i(\hat{x},\hat{y}), (L_i(\hat{x},\hat{y}))^{\perp}$ one has

$$[\nabla_x F(\hat{x},\hat{y}) + \nabla_y F(\hat{x},\hat{y})\, P_i(\hat{x})]\, h \in (L_i(\hat{x},\hat{y}))^{\perp}\,, \qquad P_i(\hat{x})h \in L_i(\hat{x},\hat{y}),$$

as well. Thus, $P_i(\hat{x})$ satisfies for each $h \in \mathbb{R}^n$ the generalized equation (2.16), and according to Corollary 2.6, $P_i(\hat{x})h$ is the unique solution of the generalized equation (2.15) with (x,y) being replaced by $(\hat{x},\hat{y})$.

As the set of all possible limits of $\nabla\tilde{S}(x_j)$, $x_j \to \hat{x}$, cannot contain other operators than the matrices $P_i(\hat{x})$ for $i \in \mathbb{K}(\hat{x})$, we see that

$$\partial\tilde{S}(\hat{x}) \subset \operatorname{conv}\{P_i(\hat{x}) \,|\, i \in \mathbb{K}(\hat{x})\}$$

by definition. It remains to observe that

$$\operatorname{conv}\left\{(P_i(\hat{x}))^T \;|\, i \in \mathbb{K}(\hat{x})\right\}\; \nabla_y f(\hat{x},\hat{y}) = \operatorname{conv}\left\{(P_i(\hat{x}))^T \nabla_y f(\hat{x},\hat{y}) \,|\, i \in \mathbb{K}(\hat{x})\right\},$$

and to realize that according to Corollary 2.6 all adjoint generalized equations (2.14) possess unique solutions $\hat{p}_i$, $i \in \mathbb{K}(\hat{x})$. Consequently, using the same arguments as in the proof of Theorem 2.3, one obtains

$$(P_i(\hat{x}))^T \nabla_y f(\hat{x},\hat{y}) = (\nabla_x F(\hat{x},\hat{y}))^T \hat{p}_i \quad \text{for each } i \in \mathbb{K}(\hat{x}),$$

and the proof is complete. □

Remark. From the course of the above proof it is clear that if we replace (H) by a weaker assumption (SRC) and assume additionally that $\nabla_y F(\hat{x}, \hat{y})$ is nonsingular on all subspaces $L_i(\hat{x}, \hat{y})$, $i \in \mathbb{K}(\hat{x})$, then the assertion of Theorem 2.7 remains true. Hypothesis (H) is frequently satisfied in discretized shape optimization problems in which due to various mechanical principles the matrices $\nabla_y F(x, y)$ are positive definite for each $x \in A$, $y \in \mathbb{R}^m$. In the variational inequalities that describe the Wardrop and the Nash equilibria, the satisfaction of (SRC) or (H) may be ensured by suitable convexity and monotonicity assumptions imposed on some problem functions [11].

Unfortunately, even in special cases when Θ happens to be convex and U_{ad} is convex, the necessary optimality conditions of Theorem 2.7 are generally not sufficient; indeed, we construct an *outer* approximation of the generalized Jacobian $\partial \tilde{S}(\hat{x})$ and thus only an outer approximation of $\partial\Theta(\hat{x})$. However, it does not concern the conditions of Theorem 2.3.

PROPOSITION 2.8. *Consider problem* (1.2) *in which* $m = n$, $f(x, y) = f_1(x) + f_2(y)$, $F(x, y) = x - Ay$, *where* A *is a given* $[n \times n]$ *matrix, and* $\Gamma = a - \mathbb{R}^n_+$, *where* $a \in \mathbb{R}^n$ *is given. Assume that:*

(i) *functions* f_1, f_2 *are convex and* f_2 *is nondecreasing with respect to the nonpositive orthant* $\mathbb{R}^n_-$; *and*

(ii) A *is an* M*-matrix* (*cf.* [32]).

Then the map S *defined by the appropriate variational inequality is single valued and the corresponding function* Θ *is convex.*

Proof. The single-valuedness of S under the assumptions imposed has been shown in [6]. In this work it has been also proved that at a fixed $x \in \mathbb{R}^n$, $S(x)$ is the maximal element of the set

$$\Omega(x) = \{y \in \mathbb{R}^m | Ay - x \leq 0,\ y - a \leq 0\}.$$

Consider two points $x_1, x_2 \in \mathbb{R}^n$ and a number $\lambda \in [0, 1]$, and denote $y_1 = S(x_1), y_2 = S(x_2)$. Clearly

$$\lambda y_1 + (1 - \lambda) y_2 \in \Omega(\lambda x_1 + (1 - \lambda) x_2),$$

because $y_1 \in \Omega(x_1)$ and $y_2 \in \Omega(x_2)$. Consequently,

$$\lambda y_1 + (1 - \lambda) y_2 - S(\lambda x_1 + (1 - \lambda) x_2) \leq 0$$

so that the operator S is convex with respect to the ordering introduced by $\mathbb{R}^n_-$. By use of this property we observe that

$$\begin{aligned}\Theta(\lambda x_1 + (1 - \lambda) x_2) = f_1(\lambda x_1 + (1 - \lambda) x_2) + f_2 \circ S(\lambda x_1 + (1 - \lambda) x_2) \leq \lambda f_1(x_1) \\ + (1 - \lambda) f_1(x_2) + f_2(\lambda y_1 + (1 - \lambda) y_2 + d)\end{aligned}$$

for some $d \in \mathbb{R}^n_-$. Therefore, as f_2 is nondecreasing with respect to $\mathbb{R}^n_-$,

$$\begin{aligned}\Theta(\lambda x_1 + (1 - \lambda) x_2) &\leq \lambda_1 f_1(x_1) + (1 - \lambda) f_1(x_2) + f_2(\lambda y_1 + (1 - \lambda) y_2) \\ &\leq \lambda \Theta(x_1) + (1 - \lambda) \Theta(x_2). \qquad \square\end{aligned}$$

Thus, if U_{ad} is convex and the assumptions of Proposition 2.8 are fulfilled, any pair $(\hat{x}, \hat{y})$ satisfying the optimality conditions of Theorem 2.3 is a solution of the appropriate problem (1.2). Certainly, one could find also different classes of problems (1.2) with this property, but such investigation goes beyond the aims of the present paper.

We conclude this section with a simple example illustrating the application of the above conditions.

Example 2.1. Consider problem (1.2) with the following variational inequality: For a given vector $x_0 \in \mathbb{R}^n$, find a pair $(z_0, w_0) \in \Gamma_1 \times \Gamma_2$ such that

$$\langle (\nabla_z g_1(x_0, z_0, w_0), \nabla_w g_2(x_0, z_0, w_0)), (z - z_0, w - w_0) \rangle \geq 0 \quad \text{for all } (z, w) \in \Gamma_1 \times \Gamma_2, \tag{2.17}$$

where the functions g_1, g_2 $[\mathbb{R}^n \times \mathbb{R}^{m_1} \times \mathbb{R}^{m_2} \longrightarrow \mathbb{R}]$ are assumed to be twice-continuously differentiable and

$$\Gamma_1 = \{v \in \mathbb{R}^{m_1} \mid C_1 v \leq d_1\},$$
$$\Gamma_2 = \{v \in \mathbb{R}^{m_2} \mid C_2 v \leq d_2\}$$

for given matrices $C_1[s_1 \times m_1]$, $C_2[s_2 \times m_2]$ and vectors $d_1 \in \mathbb{R}^{s_1}$, $d_2 \in \mathbb{R}^{s_2}$. Furthermore, we suppose that the functions $g_1(x, \cdot, \cdot)$, $g_2(x, \cdot, \cdot)$ are convex on $\Gamma_1 \times \Gamma_2$ for all $x \in \mathbb{R}^n$ and the sets Γ_1, Γ_2 are compact.

The solutions of variational inequality (2.17) are Nash equilibria in a noncooperative game of two players with the utilities $-g_1$, $-g_2$ and the sets of admissible strategies Γ_1, Γ_2. The parameter x influences both utilities so that the aim of the corresponding optimization problem (1.2) is to find a triple $(\hat{x}, \hat{z}, \hat{w}) \in \mathbb{R}^n \times \Gamma_1 \times \Gamma_2$ consisting of the vector $\hat{x}$ and a corresponding Nash equilibrium $(\hat{z}, \hat{w})$ such that a given objective f $[\mathbb{R}^n \times \mathbb{R}^{m_1} \times \mathbb{R}^{m_2} \longrightarrow \mathbb{R}]$ is minimized. Economic problems of this kind are discussed, for example, in [10].

We modify the general notation previously introduced as follows. At the optimal solution $(\hat{x}, \hat{z}, \hat{w})$ let I_1, J_1 denote the index set of active and strongly active inequalities in the constraint $z \in \Gamma_1$, respectively, and analogously let I_2, J_2 be the index sets of active and strongly active inequalities in the constraint $w \in \Gamma_2$. Furthermore,

$$\mathcal{P}((I_1, I_2) \setminus (J_1, J_2)) = \{(T_i^1, T_i^2) \mid i \in \mathbb{K}(\hat{x})\},$$

where $\mathbb{K}(\hat{x})$ is a suitable index set. A direct application of Theorem 2.7 provides us immediately with the following optimality conditions.

PROPOSITION 2.9. *Let $(\hat{x}, \hat{z}, \hat{w})$ be a (locally) optimal solution of problem* (1.2) *with the variational inequality* (2.17). *Suppose that the matrix*

$$B = \begin{bmatrix} \nabla^2_{zz} g_1(\hat{x}, \hat{z}, \hat{w}), & \nabla^2_{zw} g_1(\hat{x}, \hat{z}, \hat{w}) \\ \nabla^2_{wz} g_2(\hat{x}, \hat{z}, \hat{w}), & \nabla^2_{ww} g_2(\hat{x}, \hat{z}, \hat{w}) \end{bmatrix}$$

is positive definite and that the vectors $({}^1\hat{p}_i, {}^2\hat{p}_i) \in L_i^1(\hat{x}, \hat{z}, \hat{w}) \times L_i^2(\hat{x}, \hat{z}, \hat{w})$, $i \in \mathbb{K}(\hat{x})$, satisfy the adjoint generalized equations

$$B^T \begin{bmatrix} {}^1p_i \\ {}^2p_i \end{bmatrix} + \begin{bmatrix} \nabla_z f(\hat{x}, \hat{z}, \hat{w}) \\ \nabla_w f(\hat{x}, \hat{z}, \hat{w}) \end{bmatrix} \in \begin{bmatrix} (L_i^1(\hat{x}, \hat{z}, \hat{w}))^\perp \\ (L_i^2(\hat{x}, \hat{z}, \hat{w}))^\perp \end{bmatrix}, \tag{2.18}$$

whereby

$$L_i^1(\hat{x}, \hat{z}, \hat{w}) = \left\{v \in \mathbb{R}^{m_1} \mid \langle C_1^j, v \rangle = 0 \text{ for } j \in T_i^1 \cup J_1\right\},$$
$$L_i^2(\hat{x}, \hat{z}, \hat{w}) = \left\{v \in \mathbb{R}^{m_2} \mid \langle C_2^j, v \rangle = 0 \text{ for } j \in T_i^2 \cup J_2\right\}.$$

Then one has

$$0 \in \nabla_x f(\hat{x},\hat{z},\hat{w}) + \left[\left(\nabla^2_{zx} g_1(\hat{x},\hat{z},\hat{w})\right)^T \left(\nabla^2_{wx} g_2(\hat{x},\hat{z},\hat{w})\right)^T\right] \operatorname{conv}\left\{\left({}^1\hat{p}_i, {}^2\hat{p}_i\right) \mid i \in \mathbb{K}(\hat{x})\right\}. \tag{2.19}$$

Remark. To satisfy the assumption (H) at $(\hat{x},\hat{z},\hat{w})$, it suffices to require that $\langle v, Bv\rangle > 0$ for all $v = (v_1, v_2) \in \mathbb{R}^{m_1} \times \mathbb{R}^{m_2}$ such that $v \neq 0$, $\langle C_1^i, v_1\rangle = 0$ for $i \in J_1$, and $\langle C_2^i, v_2\rangle = 0$ for $i \in J_2$.

3. A numerical approach. Throughout this section it is assumed that all assumptions of §1 are satisfied and (SRC) holds *at all* $x \in A$, $y \in S(x)$. Additionally, we suppose that for all $x \in A$, $F(x,\cdot)$ is pseudomonotone over Γ, i.e., for all $y, v \in \Gamma$

$$\langle F(x,y), v - y\rangle \geq 0 \quad \text{implies} \quad \langle F(x,v), v - y\rangle \geq 0.$$

This assumption is not too restrictive. In some important problems, $F(x,\cdot)$ is even monotone or strongly monotone over Γ [19], [20]. It implies generally that the map S is convex valued on A (cf. [11, Proposition. 3.1]) and, thus, in our particular case, S is single valued on A due to the global validity of (SRC). Consequently, problem (1.2) can be rewritten in the form

$$\begin{aligned} &\text{minimize} && \Theta(x) := f(x, S(x)) \\ &\text{subject to} && x \in U_{\text{ad}}, \end{aligned} \tag{3.1}$$

where, for the sake of simplicity, the notation Θ is preserved for the appropriate composite function. Problem (3.1) can be solved numerically by existing NDO methods, such as the bundle methods described in [16] or [28]. Both these methods construct and update during the iteration process piecewise-affine local models of the objective which are based on the objective values and arbitrary vectors from its Clarke's generalized gradients at the single iteration points. They basically require the minimized functions to be locally Lipschitzian and directionally differentiable. In our case, Θ clearly possesses both these properties, being a composition of a continuously differentiable function and a locally Lipschitz directionally differentiable operator. However, one has to be able to compute an arbitrary vector from $\partial\Theta(x)$ for each $x \in U_{\text{ad}}$, so we concentrate on this task in the next part of this section.

For the sake of simplicity it is supposed that Γ is given merely by inequality constraints, i.e.,

$$\Gamma = \{y \in \mathbb{R}^m \mid A_2 y \leq b_2\}. \tag{3.2}$$

However, the incorporation of equality constraints does not represent any principal difficulties.

Let $x_0 \in U_{\text{ad}}$, $y_0 = S(x_0)$,

$$L_J(x_0,y_0) = \left\{z \in \mathbb{R}^m \mid \langle A_2^i, z\rangle = 0,\ i \in J(x_0,y_0)\right\},$$

$\mathcal{B}(x_0,y_0) = I(x_0,y_0)\backslash J(x_0,y_0)$, and o, o_1, o_2 be the cardinalities of $I(x_0,y_0)$, $J(x_0,y_0)$, and $\mathcal{B}(x_0,y_0)$, respectively. Denote by A_2^I the $[o \times m]$ matrix composed from the rows A_2^i, $i \in I(x_0,y_0)$. Evidently, A_2^I may be divided into two matrices A_2^J and $A_2^{\mathcal{B}}$ composed from the rows A_2^i for $i \in J(x_0,y_0)$ and $i \in \mathcal{B}(x_0,y_0)$, respectively.

PROPOSITION 3.1. *Assume that the linear system*

$$-\left(A_2^{\mathcal{B}}\right)^T y_1^* + \left(\nabla_y F(x_0,y_0)\right)^T y_2^* + \left(A_2^J\right)^T y_3^* = 0 \tag{3.3}$$

does not possess a solution $(y_1^*, y_2^*, y_3^*) \in \mathbb{R}^{o_2} \times \mathbb{R}^m \times \mathbb{R}^{o_1}$ *satisfying the conditions*

$$\begin{gathered} y_1^* \geq 0, \qquad y_1^* \neq 0, \\ y_2^* \in \mathcal{N}\left(\left(\nabla_x F(x_0,y_0)\right)^T\right) \cap \mathcal{N}\left(A_2^J\right). \end{gathered} \tag{3.4}$$

Furthermore, let $\nabla_y F(x_0,y_0)$ *be nonsingular on* $\mathcal{N}(A_2^J)$. *Then the operator* $P_J(x_0)$ *that assigns to an arbitrary vector* $h \in \mathbb{R}^n$ *the (unique) solution* v *of the linear generalized equation*

$$0 \in \nabla_x F(x_0,y_0)h + \nabla_y F(x_0,y_0)v + N_{L_J(x_0,y_0)}(v) \tag{3.5}$$

belongs to $\partial S(x_0)$.

Proof. We start with an observation that $P_J(x_0)$ is uniquely determined by (3.5) due to the nonsingularity of $\nabla_y F(x_0,y_0)$ on $\mathcal{N}(A_2^J)$.

For Γ given by (3.2) it is clear from Proposition 2.2 that for $x \in A$, $y = S(x)$ the strict complementarity condition $I(x,y) = J(x,y)$ ensures the differentiability of S at x (cf. also [11, Theorem 5.3]). Thus, taking into account the Lipschitz continuity of S, it suffices to show the existence of a direction $h \in \mathbb{R}^n$ such that

$$J(x_0 + \vartheta h, S(x_0 + \vartheta h)) = I(x_0 + \vartheta h, S(x_0 + \vartheta h)) = J(x_0,y_0) \tag{3.6}$$

for all sufficiently small positive ϑ. Indeed, if this were the case, then there would exist a sequence $\vartheta_i \downarrow 0$ such that S is differentiable at the points $x_0 + \vartheta_i h$, $i = 1,2,\ldots$, and at the same time $\nabla S(x_0 + \vartheta_i h) \longrightarrow P_J(x_0)$ (cf. the proof of Theorem 2.7). By definition, one has $P_J(x_0) \in \partial S(x_0)$.

On the basis of the assertion of Lemma 2.4, the validity of condition (3.6) is guaranteed by the inequality

$$A_2^{\mathcal{B}} S'(x_0;h) < 0, \tag{3.7}$$

because then for all sufficiently small positive ϑ the constraints $\langle A_2^i, S(x_0 + \vartheta h)\rangle \leq b_2^i$, $i \in \mathcal{B}(x_0,y_0)$, are not active. The directional derivative $S'(x_0;h)$ is given by the generalized equation (2.2), where

$$\begin{aligned} K(x_0,y_0) &= \left\{v \in \mathbb{R}^m \mid \langle F(x_0,y_0), v\rangle = 0,\ \langle A_2^i, v\rangle \leq 0 \text{ for } i \in I(x_0,y_0)\right\} \\ &= \left\{v \in \mathbb{R}^m \mid \langle A_2^i, v\rangle = 0 \text{ for } i \in J(x_0,y_0),\ \langle A_2^i, v\rangle \leq 0 \text{ for } i \in \mathcal{B}(x_0,y_0)\right\} \end{aligned}$$

due to (2.9). Consequently

$$N_{K(x_0,y_0)}(v) = \left\{ z \in \mathbb{R}^m \,|\, z = \sum_{i \in J(x_0,y_0)} \alpha^i A_2^i + \sum_{i \in \mathcal{B}(x_0,y_0)} \beta^i A_2^i, \right. \\ \left. \text{where } \beta^i \geq 0 \text{ and } \beta^i \langle A_2^i, v\rangle = 0,\ i \in \mathcal{B}(x_0,y_0) \right\}.$$

Therefore, by combining (3.7) and (2.2), we may conclude that the existence of the desired direction h is implied by the existence of a multiplier vector $\alpha \in \mathbb{R}^{o_1}$ such

that the following linear system of equalities and inequalities

$$\begin{aligned} \nabla_x F(x_0,y_0)h + \nabla_y F(x_0,y_0)v + \left(A_2^J\right)^T \alpha &= 0, \\ A_2^J v &= 0, \\ A_2^{\mathcal{B}} v &< 0 \end{aligned} \tag{3.8}$$

is consistent. It remains to apply the well-known Motzkin theorem of the alternative [17], according to which the consistency of (3.8) is equivalent to the inconsistency of (3.3), (3.4). □

COROLLARY 3.2. *Assume that* $\nabla_x F(x_0,y_0)$ *is surjective and* $\nabla_y F(x_0,y_0)$ *is non-singular on* $\mathcal{N}(A_2^J)$. *Then the assertion of Proposition* 3.1 *holds true.*

Proof. By the assumption one has $\mathcal{N}\left(\nabla_x F(x_0,y_0))^T\right) = \{0\}$ and hence $y_2^* = 0$. Condition (LI) then implies that there does not exist any nonzero y_1^* satisfying (3.3). □

The proposed approach has been numerically tested on simple examples of the type (1.2) with the following variational inequality:

For a given vector $x_0 \in A$, find a pair $(u_0,\eta_0) \in \mathbb{R}^m \times \mathbb{R}^s_+$ such that

$$\begin{aligned} \langle (\nabla_v \mathcal{L}(x_0,u_0,\eta_0), -\nabla_\eta \mathcal{L}(x_0,u_0,\eta_0)),\ (u-u_0,\eta-\eta_0)\rangle \geq 0 \\ \text{for all } (u,\eta) \in \mathbb{R}^m \times \mathbb{R}^s_+, \end{aligned} \tag{3.9}$$

where $\mathcal{L}(x_0,\cdot,\cdot)$ is the standard Lagrangian of the optimization problem

$$\begin{aligned} &\text{minimize} && \varphi(x_0,u) \\ &\text{subject to} && \psi^i(x_0,u) \leq 0, \qquad i = 1,2,\ldots,s, \end{aligned} \tag{3.10}$$

i.e.,

$$\mathcal{L}(x_0,u,\eta) = \varphi(x_0,u) + \sum_{i=1}^{s} \eta^i \psi^i(x_0,u). \tag{3.11}$$

We assume that functions φ, ψ^i, $i = 1,2,\ldots,s$, are twice-continuously differentiable on $A \times \mathbb{R}^m$ and convex in the second variable and denote

$$\begin{aligned} M(x,u) &= \left\{i \in \{1,2,\ldots,s\} \mid \psi^i(x,u) = 0\right\}, \\ \Xi:\ x &\longmapsto \left\{\operatorname{argmin} \varphi(x,u) \mid \psi^i(x,u) \leq 0,\ i = 1,2,\ldots,s\right\}. \end{aligned}$$

Furthermore, it is supposed that for all $x_0 \in A$ one has $\Xi(x_0) \neq \emptyset$, and the gradients $\nabla_u \psi^i(x_0,u_0), i \in M(x_0,u_0)$, are linearly independent whenever $u_0 \in \Xi(x_0)$. Then to any $x_0 \in A$, $u_0 \in \Xi(x_0)$ we can assign the unique multiplier vector $\eta_0 \in \mathbb{R}^s_+$, satisfying the appropriate K. K. T. conditions. Finally, we impose still the standard strong second-order sufficient condition on A.

(SSOSC) for all $x_0 \in A$, $u_0 \in \Xi(x_0)$, η_0 being the corresponding K. K. T. vector and $z \in \mathbb{R}^m \backslash \{0\}$, one has $\langle z,\ \nabla^2_{uu} \mathcal{L}(x_0,u_0,\eta_0) z\rangle > 0$, whenever

$$\langle \nabla_u \psi^i(x_0,u_0), z\rangle = 0 \quad \text{for } i \in N(x_0,u_0) := \left\{j \in M(x_0,u_0) \mid \eta_0^j > 0\right\}.$$

As pointed out in [25, Proposition 4.1], under the above assumptions (SRC) holds for (3.9) at any $(x_0,u_0,\eta_0), x_0 \in A$, and (u_0,η_0) being the appropriate solution to (3.9).

One immediately observes that the variational inequality (3.9) assigns to admissible parameters $x_0 \in U_{\rm ad}$ (unique) K. K. T. points of the inner convex programs (3.10). In the corresponding problem (1.2) we thus optimize the location of the saddle points of the Lagrangian (3.11) over $\mathbb{R}^m \times \mathbb{R}^s_+$ with respect to the given (upper level) objective f, dependent on (x, u, η). A vector from the generalized gradient of Θ (cf. (3.1)) can be computed as follows.

PROPOSITION 3.3. *Let $x_0 \in U_{\rm ad}$ be given and $(u_0, \eta_0) = S(x_0)$ be the corresponding solution of variational inequality* (3.9). *Assume that the pair $(p_1, p_2) \in \mathbb{R}^m \times \mathbb{R}^s$ solves the adjoint linear system*

$$\begin{aligned} &\nabla^2_{uu}\mathcal{L}(x_0, u_0, \eta_0)\, p_1 - \sum_{i \in M(x_0,u_0)} p_2^i \nabla_u \psi^i(x_0, u_0) + \nabla_u f(x_0, u_0, \eta_0) = 0, \\ &\big\langle \nabla_u \psi^i(x_0, u_0), p_1 \big\rangle + \big(\nabla_\eta f(x_0, u_0, \eta_0)\big)^i = 0 \quad \text{for } i \in M(x_0, u_0) \\ &p_2^i = 0 \quad \text{for } i \in \{1, 2, \ldots, s\} \setminus M(x_0, u_0). \end{aligned} \tag{3.12}$$

Then one has

$$\nabla_x f(x_0, u_0, \eta_0) + \big(\nabla^2_{ux}\mathcal{L}(x_0, u_0, \eta_0)\big)^T p_1 - \big(\nabla^2_{\eta x}\mathcal{L}(x_0, u_0, \eta_0)\big)^T p_2 \in \partial\Theta(x_0), \tag{3.13}$$

whenever the inconsistency assumption of Proposition 3.1 *holds.*

Proof. Clearly,

$$J(x_0, u_0, \eta_0) = \big\{ i \in \{1, 2, \ldots, s\} \,|\, \psi^i(x_0, u_0) < 0 \big\} = \{1, 2, \ldots, s\} \setminus M(x_0, u_0)$$

so that

$$L_J(x_0, u_0, \eta_0) = \big\{ (z_1, z_2) \in \mathbb{R}^m \times \mathbb{R}^s \,|\, z_2^i = 0 \quad \text{for } i \in \{1, 2, \ldots, s\} \setminus M(x_0, u_0) \big\}.$$

It implies that for the solution v of the generalized equation (3.5) one has

$$v = (v_1, v_2) \in \mathbb{R}^m \times \mathbb{R}^s \quad \text{with } v_2^i = 0 \quad \text{for } i \in \{1, 2, \ldots, s\} \setminus M(x_0, u_0)$$

and the condition $v^\star \in N_{L_J(x_0,u_0,\eta_0)}(v)$ yields $v^\star = (v_1^\star, v_2^\star) \in \mathbb{R}^m \times \mathbb{R}^s$ with $v_1^\star = 0$ and $v_2^{\star i} = 0$ for $i \in M(x_0, u_0)$. Thus, (3.5) may be written in the substantially reduced form

$$\begin{aligned} 0 &= \begin{bmatrix} \nabla^2_{ux}\mathcal{L}(x_0, u_0, \eta_0) \\ -H(x_0, u_0) \end{bmatrix} h + \begin{bmatrix} \nabla^2_{uu}\mathcal{L}(x_0, u_0, \eta_0) & (G(x_0, u_0))^T \\ -G(x_0, u_0) & 0 \end{bmatrix} \begin{bmatrix} v_1 \\ \tilde{v}_2 \end{bmatrix}, \\ v_2^i &= 0 \qquad \text{for } i \in \{1, 2, \ldots, s\} \setminus M(x_0, u_0), \end{aligned} \tag{3.14}$$

where $\tilde{v}_2$ is composed from the coordinates v_2^i for $i \in M(x_0, u_0)$ and matrices $H(x_0, u_0)$, $G(x_0, u_0)$ are composed from the rows $\nabla_x \psi^i(x_0, u_0)$, $\nabla_u \psi^i(x_0, u_0)$ for $i \in M(x_0, u_0)$, respectively. The linear independence of the gradients $\nabla_u \psi^i(x_0, u_0)$ together with (SSOSC) imply immediately that the matrix

$$\begin{bmatrix} \nabla^2_{uu}\mathcal{L}(x_0, u_0, \eta_0) & (G(x_0, u_0))^T \\ -G(x_0, u_0) & 0 \end{bmatrix}$$

is nonsingular (which means that the corresponding operator $\nabla_y F(x_0, y_0)$ is nonsingular on $L_J(x_0, y_0)$). Thus, in view of Proposition 3.1, the operator $P_J(x_0)$, assigning to an arbitrary vector $h \in \mathbb{R}^n$ the unique solution (v_1, v_2) of (3.14), belongs to $\partial S(x_0)$.

TABLE 3.1
Objectives.

Example	$f(x,u,\eta)$
3.1	$\frac{1}{2}(u^1-3.0)^2+\frac{1}{2}(u^2-4.0)^2$
3.2	$\frac{1}{2}(u^1-3.0)^2+\frac{1}{2}(u^2-4.0)^2+\frac{1}{2}(\eta^1-1.0)^2$
3.3	$\frac{1}{2}(u^1-3.0)^2+\frac{1}{2}(u^2-4.0)^2+5(\eta^2)^2$
3.4	$\frac{1}{2}x^2+\frac{1}{2}(u^1-3.0)^2+\frac{1}{2}(u^2-4.0)^2+\frac{1}{2}(\eta^1-1.0)^2+\frac{1}{2}(\eta^2-1.0)^2$

TABLE 3.2
Results with the starting point $x = 0.0$.

Example	Optimal Objective Value	Optimal Solution					Number of Iterations
		x	u^1	u^2	η^1	η^2	
3.1	3.2077	4.0604	2.6822	1.4871	0.0	0.6621	9
3.2	3.4494	5.1535	2.7487	1.4	0.7252	0.8239	9
3.3	4.6043	2.3893	2.7893	1.2077	0.0	0.3697	10
3.4	6.5927	1.3733	2.8875	0.8942	0.0	0.1991	14

It remains to express the vector $(P_J(x_0))^T(\nabla_u f(x_0,u_0,\eta_0),\nabla_\eta f(x_0,u_0,\eta_0))$ using the solution of the adjoint generalized equation (2.14) corresponding to variational inequality (3.9) and the index set $J(x_0,u_0,\eta_0)$. Evidently, this adjoint generalized equation attains the form (3.12) because $\nabla^2_{uu}\mathcal{L}(x_0,u_0,\eta_0)$ is symmetric. Thus, as in the proof of Theorem 2.3, one obtains

$$\begin{aligned}&(P_J(x_0))^T(\nabla_u f(x_0,u_0,\eta_0),\nabla_\eta f(x_0,u_0,\eta_0))\\&\quad=\left(\nabla^2_{ux}\mathcal{L}(x_0,u_0,\eta_0)\right)^T p_1-\left(\nabla^2_{\eta x}\mathcal{L}(x_0,u_0,\eta_0)\right)^T p_2\end{aligned}$$

with (p_1,p_2) being the (unique) solution of (3.12). □

We have numerically solved four simple examples of the above type. In all of them $n=1$, $m=2$, $s=4$, $U_{\text{ad}}=[0,10]$, and problem (3.10) has the following form:

$$\begin{aligned}&\text{minimize} && \tfrac{1}{2}\left[(1+0.2x)(u^1)^2+(1+0.1x)(u^2)^2\right]-(3+1.333x)u^1-xu^2\\&\text{subject to} && -0.333u^1+u^2-1+0.1x\le 0,\\&&& (u^1)^2+(u^2)^2-9-0.1x\le 0,\\&&& -u^1\le 0,\\&&& -u^2\le 0.\end{aligned}\tag{3.15}$$

They differ only in the (upper level) objective f. In all examples we have used the NDO code BT described in [28] with the accuracy being specified by $\varepsilon=0.0001$ and two different starting points $x=0.0$ and $x=10.0$.

The single objectives f are collected in Table 3.1 and the results are summarized in Tables 3.2 and 3.3.

At the solution of Example 3.1 the first inequality in (3.15) is semiactive, and indeed, the corresponding function Θ is nondifferentiable at $\hat{x}=4.0604$ ($\Theta'(\hat{x};1)=0.1791$, $\Theta'(\hat{x};-1)=0.2855$). As the vector from $\partial\Theta(\hat{x})$, computed according to Proposition 3.3, is strictly nonzero, a classical "smooth" minimization method, testing in its stopping criterion a norm of the gradient, could probably not terminate at $\hat{x}$.

All computations have been performed on a VAX 6000–310 computer and the CPU times have not exceeded 3 seconds. For the solution of problems (3.10) the

TABLE 3.3
Results with the starting point $x = 10.0$.

Example	Optimal Objective Value	Optimal Solution					Number of Iterations
		x	u^1	u^2	η^1	η^2	
3.1	3.2077	4.0604	2.6822	1.4871	0.0	0.6621	6
3.2	3.4494	5.1535	2.7487	1.4	0.7252	0.8239	8
3.3	4.6043	2.3906	2.7892	1.208	0.0	0.37	8
3.4	6.5927	1.3733	2.8875	0.8943	0.0	0.1992	14

sequential quadratic programming code of [27] has been used. It has provided us also with the K. K. T. vector η; i.e., it has completely solved the variational inequality (3.9).

Remark. Example 3.1 has been solved in [24] in a slightly different way.

Conclusion. Under the assumptions imposed, the optimality conditions of §2 properly reflect the structure of (1.2), taking into account the special properties of $S(\tilde{S})$. Expectantly, the semiactive inequality constraints play here a similar role as in the class of Stackelberg problems, studied in [24]. These constraints have a very lucid interpretation in discretized contact problems of mechanics: At a node of discretization, where an elastic body is in contact with a rigid obstacle, the appropriate (nonpenetration) constraint is semiactive, provided no reaction force of the obstacle at this node is needed to keep the system in equilibrium.

If $\Gamma = \mathbb{R}^m$, $S(\tilde{S})$ happens to be a continuously differentiable implicit map and our conditions attain a classical "optimal control" form (in the finite-dimensional setting).

To the numerical approach of §3 let us remark that it is evidently not possible to verify the inconsistency assumption of Proposition 3.1 during the iteration process. Nevertheless, our experience from the numerical solution of medium-sized Stackelberg problems by principally the same approach (cf. [23] and [15]) shows its effectiveness and reliability even without this verification. It deserves further testing on problems of a suitable size.

Acknowledgment. The author would like to thank the referees for their careful review of the manuscript and for the helpful comments and constructive suggestions that they provided.

REFERENCES

[1] G. ANANDALINGAM AND T. L. FRIESZ, EDS., *Hierarchical optimization*, Ann. Oper. Res., 34 (1992), pp. 1–331.
[2] J.-P. AUBIN AND I. EKELAND, *Applied Nonlinear Analysis*, J. Wiley & Sons, New York, 1984.
[3] J.-P. AUBIN AND H. FRANKOWSKA, *Set-valued Analysis*, Birkhäuser, Boston, 1990.
[4] F. H. CLARKE, *Optimization and Nonsmooth Analysis*, J. Wiley & Sons, New York, 1983.
[5] R. W. COTTLE, J. KYPARISIS, AND J. S. PANG, *Foreword to a special issue devoted to variational inequality problems*, Math. Programming, 48 (1990), pp. iii–iv.
[6] J. F. DURAND, *L'algorithme de Gauss-Seidel apliqué à un problème unilatéral non symétrique*, RAIRO, R-2 (1972), pp. 23–30.
[7] T. L. FRIESZ, R. L. TOBIN, AND T. MILLER, *Existence theory for spatially competitive network facility location models*, Ann. Oper. Res., 18 (1989), pp. 267–276.
[8] T. L. FRIESZ, R. L. TOBIN, H.-J. CHO, AND N. J. MEHTA, *Sensitivity analysis based heuristic algorithms for mathematical programs with variational inequality constraints*, Math. Programming, 48 (1990), pp. 265–284.

[9] T. L. FRIESZ, HSUN-JUNG CHO, N. J. MEHTA, R. L. TOBIN, AND G. ANANDALINGAM, *A simulated annealing approach to the network design problem with variational inequality constraints*, Transportation Sci., 26 (1992), pp. 18–26.

[10] P. T. HARKER AND S. C. CHOI, *A penalty function approach for mathematical programs with variational inequality constraints*, WP 87-09-08, Decision Sciences Department, Univ. of Pennsylvania, Philadelphia, PA, 1987.

[11] P. T. HARKER AND J. S. PANG, *Finite-dimensional variational inequality and nonlinear complementarity problems: A survey of theory, algorithms and applications*, Math. Programming, 48 (1990), pp. 161–220.

[12] J. HASLINGER AND P. NEITTAANMÄKI, *Finite Element Approximation for Optimal Shape Design: Theory and Applications*, J. Wiley & Sons, Chichester, UK, 1988.

[13] J.-B. HIRIART-URRUTY, *Tangent cones, generalized gradients and mathematical programming in Banach spaces*, Math. Oper. Res., 4 (1979), pp. 79–97.

[14] J. KYPARISIS, *Solution differentiability for variational inequalities*, Math. Programming, 48 (1990), pp. 285–301.

[15] M. KOČVARA AND J. V. OUTRATA, *A nondifferentiable approach to the solution of optimum design problems with variational inequalities*, in System Modelling and Optimization, P. Kall, ed., Lecture Notes in Control and Inform. Sci., Springer-Verlag, Berlin, 1992, pp. 364-373.

[16] C. LEMARÉCHAL, J.-J. STRODIOT, AND A. BIHAIN, *On a bundle algorithm for nonsmooth optimization*, in Nonlinear Programming 4, O. L. Mangasarian, R. R. Meyer, and S. M. Robinson, eds., Academic Press, New York, 1981.

[17] O. L. MANGASARIAN, *Nonlinear Programming*, McGraw Hill, New York, 1969.

[18] P. MARCOTTE, *Network design problem with congestion effects: A case of bilevel programming*, Math. Programming, 34 (1986), pp. 142–162.

[19] F. MIGNOT, *Contrôle dans les inéquations variationelles elliptiques*, J. Funct. Anal., 22 (1976), pp. 130–185.

[20] F. MIGNOT AND J.-P. PUEL, *Optimal control in some variational inequalities*, SIAM J. Control Optim., 22 (1984), pp. 466–476.

[21] T. MILLER, R. TOBIN, AND T. L. FRIESZ, *Stackelberg games on a network with Cournot-Nash oligopolistic competitors*, J. Regional Sci., 31 (1991), pp. 435–454.

[22] J. J. MORÉ, *Coercivity conditions in nonlinear complementarity problems*, SIAM Rev., 17 (1974), pp. 1–16.

[23] J. V. OUTRATA, *On the numerical solution of a class of Stackelberg problems*, Z. Oper. Res., 4 (1990), pp. 255–278.

[24] ———, *On necessary optimality conditions for Stackelberg problems*, J. Optim. Theory Appl., 76 (1993), pp. 305–320.

[25] S. M. ROBINSON, *Strongly regular generalized equations*, Math. Oper. Res., 5 (1980), pp. 43–62.

[26] ———, *An implicit-function theorem for a class of nonsmooth functions*, Math. Oper. Res., 16 (1991), pp. 282–309.

[27] K. SCHITTKOWSKI, *NLPQL: A Fortran subroutine solving constrained nonlinear programming problems*, Ann. Oper. Res., 5 (1985), pp. 485–500.

[28] H. SCHRAMM AND J. ZOWE, *A version of the bundle idea for minimizing a nonsmooth function: Conceptual idea, convergence analysis, numerical results*, SIAM J. Optim., 2 (1992), pp. 121–152.

[29] SHUZHONG SHI, *Optimal control of strongly monotone variational inequalities*, SIAM J. Control Optim., 26 (1988), pp. 274–290.

[30] ———, *Erratum: Optimal control of strongly monotone variational inequalities*, SIAM J. Control Optim., 28 (1990), pp. 243–249.

[31] R. TOBIN AND T. L. FRIESZ, *Spatial competition facility location models: definition, formulation and solution approach*, Ann. Oper. Res., 6 (1986), pp. 49–74.

[32] R. S. VARGA, *Matrix Iterative Analysis*, Prentice-Hall, London, 1962.

SIAM J. OPTIMIZATION
Vol. 4, No. 2, pp. 358–382, May 1994
© 1994 Society for Industrial and Applied Mathematics
008

TRIANGULAR DECOMPOSITION METHODS FOR SOLVING REDUCIBLE NONLINEAR SYSTEMS OF EQUATIONS*

J. E. DENNIS, JR.†, JOSÉ MARIO MARTÍNEZ‡, AND XIAODONG ZHANG§

Abstract. This paper generalizes to the nonlinear case a standard way to solve general sparse systems of linear equations. In particular, Duff [*J. Inst. Math. Appl.*, 19 (1977), pp. 339–342] has suggested that row and column interchanges be applied to permute the coefficient matrix of a linear system into block lower triangular form. The linear system then can be solved by using the associated block Gauss–Seidel or forward block substitution scheme. This is the approach taken in the Harwell MA28 routine. If more than one matrix with the same sparsity pattern is to be factored, then the reordering need not be redone. In extending this approach to nonlinear problems, it is necessary to assume as in the linear case that component equations can be evaluated separately from equations in other blocks. The algorithm for doing the reordering is very fast, and if the equations can be put into block lower triangular form with each block size much less than the dimension of the system, then a large savings is in order because only the diagonal blocks need to be factored. In the nonlinear variants considered here, there are additional advantages. Not only are just the diagonal blocks of the Jacobian matrix computed and factored, but the off-diagonal partial derivatives need not even exist. Numerical tests and analytic results affirm the intuition that these variants are superior locally to Newton's method. Current work is concerned with globalizing these methods and with variants especially suited to parallel implementations.

Key words. block triangular decomposition, Gauss–Seidel–Newton method, sparse nonlinear systems of equations, parallel processing

AMS subject classifications. 65H10, 65W05

1. Introduction.

This paper will consider the problem of solving the large-scale nonlinear system of equations

$$F(x) = 0, \tag{1.1}$$

where $F : R^n \to R^n$ and the Jacobian $J(x)$ is sparse. We will assume we know the structural sparsity pattern of the Jacobian matrix $J(x^k)$. By this we mean that we will assume we have encoded which components of x affect the value of each component f_i of F. We admit that there may be some incidental additional sparsity in $J(x)$ at some particular x caused by some of the partial derivatives happening to be zero at that x, but we do not attempt to exploit incidental sparsity. One way to determine structural sparsity would be to use finite differences to compute the Jacobian at some point, and assume that any exact zero in the result is a structural zero. Of course, it is possible to foil this scheme, but it is highly unlikely to fail in practice.

* Received by the editors March 16, 1992; accepted for publication (in revised form) March 16, 1993. This research was supported in part by National Science Foundation Cooperative Agreement CCR-8809615, National Science Foundation grants CCR-9008991, CCR-9047481, CCR-9102954, and DMS-8903751, Air Force Office of Scientific Research grant 89-0363, Department of Energy grant ER/25017-3, and Army Research Office grant DAAL grant 03-90-0093.

† Department of Computational and Applied Mathematics and Center for Research on Parallel Computation, Rice University, Houston, Texas 77251-1892.

‡ Department of Applied Mathematics, University of Campinas, 13081 Campinas, SP, Brazil. This work was partially conducted while the author visited the Center for Research on Parallel Computation at Rice University. The work of this author was supported by FAPESP grant 90/3724/6 and by FAEP-UNICAMP.

§ High Performance Computing and Software Laboratory, University of Texas at San Antonio, San Antonio, Texas 78249. This work was partially conducted while the author visited the Center for Research on Parallel Computation at Rice University.

We will analyze a rather large class of methods. They are decomposition methods, and they share the congenital advantages and disadvantages of this class. Among them, if a problem can be decomposed in the requisite way, it will be rather obvious that the correct choice of a method from this class would be a sensible way to attack the problem. On the other hand, the choice of a particular member of this class for a particular problem is always going to depend on some properties of the function, but we think there are many hints here as to how to make this decision. However, there are some basic assumptions we make in order to be reasonably confident that one should choose one of the methods discussed here rather than a straightforward Newton's method, which is in fact one of the methods considered here when the sparsity is sufficiently random.

The main assumption we make in order to use effectively one of the methods considered here, is that the components of F should be able to be partitioned into blocks of components that can be evaluated each at different points in a total time roughly equivalent to the evaluation of F at a single point. This is a strong assumption which is by no means always correct, but it is true for many large problems, where it is not at all unusual to have difficulty in applying library versions of Newton's method because they usually assume that the user at least will furnish a routine that accepts x and returns $F(x)$. In many cases, the user assembles and encodes the residual calculations in blocks and never thinks in terms of computing the entire vector $F(x)$ at one time. To make things simple, we make our presentation here as if each component could constitute such a block.

The second assumption is that Tarjan's algorithm [16] applied in the usual way Duff [6] to the sparsity structure of the Jacobian $F'(x)$ would result in a set of row and column permutations P and Q that would give $PF'(x)Q$ a nontrivial block lower triangular structure. This actually goes somewhat with the first assumption in practice, but when the decomposition gives the entire Jacobian as a single block, then all the methods given here reduce to Newton's method with various strategies for when to reevaluate the Jacobian.

In large engineering systems, these properties are very common. To see this, consider the simulation of a large system of roughly sequential processes like a chemical plant. It is standard in chemical engineering to have a library of equation models of component processes (distillation columns, catalytic crackers, etc.) and, for a specific design problem, to pull these off the shelf and connect them by additional equations that set the outputs from a certain process to the inputs to its daughter processes. Of course, there can be feedback loops that complicate the Jacobian structure, but that is dealt with cheaply by Tarjan's algorithm, and the resulting block lower triangular decomposition of such structures generally is found to be very useful in solving for the Newton step.

The last difficulty we mention is again common to decomposition methods. These methods generally are more difficult to implement in such a way as to ensure convergence from poor initial guesses. For example, no one has yet published a convincing way to globalize any of the Brown–Brent methods. And yet, Brown–Brent methods and Gauss–Seidel methods, which include the ones proposed here, are used regularly in practice. There are at least two reasons for this. The first is that these decomposition methods seem to be able to converge unmodified from worse initial guesses than methods that make more superficial use of structure. (See Moré and Cosnard [12].) The second is that for many applications there are very good initial guesses available as a matter of course. Still, we are investigating ways to extend the region

of convergence for these methods.

2. Newton's method. The standard Newton's method for (1.1) generates successive estimates $x^k \in R^n$ of a solution $x^* \in R^n$ of (1.1) as:

$$\text{(2.1)} \quad \text{Solve} \quad J(x^k)s^N = -F(x^k) \quad \text{and set} \quad x^{k+1} = x^k + s^N, \quad k = 0, 1, \ldots.$$

In the case of large sparse nonlinear problems, a single standard method, such as the Newton's method, may not handle all the instances of (1.1) efficiently, but rather the algorithm must take into account the sparsity structure and other special characteristics of the problem. Our purpose here is to suggest one way to use existing technology automatically to tailor methods for particular sparsity patterns.

As motivation, we will suppose we decided to use Newton's method on a general instance of (1.1). Then we probably would use a graph coloring approach to compute $J(x^k)$ efficiently (see Curtis, Powell, and Reid [4], Coleman and Moré [3], and Dennis and Schnabel [5]). After that, we might apply the Harwell code MA28 to solve the linear system (2.1). If we did so, then permutation matrices P and Q would be found such that

$$\text{(2.2)} \qquad PJ(x^k)Q(Q^T s^N) = -PF(x^k)$$

is a block lower triangular system. Of course, the system may have only one block, but the search for the permutations is cheap enough (see Harary [10], Duff [6], and Duff, Erisman, and Reid [8]) that nothing significant is lost in trying. On the other hand, if a nontrivial block triangular structure is revealed, then the Newton step can be found by forward block substitution, which is the same as a linear block Gauss–Seidel sweep down the permuted Newton system. This would require only that the diagonal blocks be factored. Of course, the diagonal blocks of $PJ(x)Q$ are nonsingular if and only if $J(x)$ is nonsingular. For many problems, the structural sparsity of $J(x)$ stays constant across all iterations and so it is not necessary to reorder at every iteration.

The foregoing is exactly the algorithm we refer to when we compare our suggested methods to Newton's method.

In a nutshell, what we propose here is to reorder the equations and variables still only once, but *before*, rather than after, starting the nonlinear solution process. If Newton's method is implemented as outlined above, then reordering information is being computed and used in the linear inner loop, but is not being used by the nonlinear outer loop. Our purpose is to argue that this is wasteful.

In the next section, we will make precise the comparison between the Newton method defined by (2.2) and the algorithms we suggest, which are variants of block Gauss–Seidel–Newton, or block Jacobi–Newton in the terminology introduced in Ortega and Rheinboldt [14]. In §3, we outline some variants, which we analyze in §4. Section 5 contains the results of some numerical experiments that show the advantage of our approach. Section 6 is devoted to a discussion and summary of these results and where they seem to point. The proofs of the convergence results are included in the Appendix.

3. Newton–Gauss–Seidel vs. Gauss–Seidel–Newton. It will simplify notation greatly if we assume for the remainder of this paper that the permutations have been absorbed into the problem. Thus, when we write x, we really mean $Q^T x$; when we write $F(x)$, we really mean $PF(Q^T x)$; and $J(x)$ is really the block lower triangular matrix $PF'(x)Q$. Thus, if we apply Newton's method to solving this incarnation of

(1.1) given by

$$F(x) = \begin{cases} F_1(x_1), \\ F_2(x_1,\ x_2), \\ \vdots \\ F_m(x_1,\ x_2, \ldots,\ x_m), \end{cases} \tag{3.1}$$

where

$$x = (x_1,\ x_2, \ldots,\ x_m)^T \in R^{n_1} \times R^{n_2} \times \cdots \times R^{n_m} = R^n,$$

$$F_i : R^{n_1} \times R^{n_2} \times \cdots \times R^{n_i} \to R^{n_i}, \qquad i = 1,\ 2, \ldots,\ m,$$

and

$$\sum_{i=1}^{m} n_i = n,$$

then we have to solve

$$\begin{cases} \frac{\partial F_1(x_1^k)}{\partial x_1} s_1^N = -F_1(x_1^k), \\ \frac{\partial F_2(x_1^k, x_2^k)}{\partial x_1} s_1^N + \frac{\partial F_2(x_1^k,\ x_2^k)}{\partial x_2} s_2^N = -F_2(x_1^k, x_2^k), \\ \vdots \\ \frac{\partial F_m(x_1^k, x_2^k, \ldots, x_m^k)}{\partial x_1} s_1^N + \frac{\partial F_m(x_1^k,\ x_2^k, \ldots, x_m^k)}{\partial x_2} s_2^N + \cdots + \frac{\partial F_m(x_1^k,\ x_2^k, \ldots, x_m^k)}{\partial x_m} s_m^N \\ \qquad = -F_m(x_1^k,\ x_2^k, \ldots, x_m^k) \end{cases} \tag{3.2}$$

and set $x_i^{k+1} = x_i^k + s_i^N$ for $i = 1, \ldots, m$ and $k = 0, 1, 2, \ldots$.

We should use forward block substitution, which is the same as the block Gauss–Seidel linear iteration, to compute the Newton step by

$$\frac{\partial F_1(x_1^k)}{\partial x_1} s_1^N = -F_1(x_1^k), \tag{3.3}$$

and for $i = 2, \ldots, m$, s_i^N comes from the $n_i \times n_i$ linear system

$$\frac{\partial F_i(x_1^k, \ldots, x_i^k)}{\partial x_i} s_i^N = -\left(F_i(x_1^k, \ldots, x_i^k) + \sum_{j=1}^{i-1} \frac{\partial F_i(x_1^k, \ldots, x_i^k)}{\partial x_j} s_j^N \right). \tag{3.4}$$

But now the key observation is that the right sides of (3.4) are just the first-order Taylor approximations to $-F_i(x_1^k + s_1^N, x_2^k + s_2^N, \ldots, x_{i-1}^k + s_{i-1}^N, x_i^k)$. Thus, if we can compute values of F_i independently, it must surely be better to do so than to build a Taylor approximation to it at the expense of computing the strict lower triangular part of the Jacobian. This leads us to suggest the block version of the Gauss–Seidel–Newton method given by

$$\frac{\partial F_i(x_1^k, \ldots, x_i^k)}{\partial x_i} s_i = -F_i(x_1^k + s_1, x_2^k + s_2, \ldots, x_{i-1}^k + s_{i-1}, x_i^k), \tag{3.5}$$

where $x_i^{k+1} = x_i^k + s_i$, $i = 1, \ldots, m$ and $k = 0, 1, 2, \ldots$.

It is worth noting here that the fact that the lower triangular parts of the Jacobian do not appear in the algorithm has the effect that the local quadratic convergence analysis given below does not require the lower triangular parts of the Jacobian even to exist. In contrast, for Newton's method, they must exist and be smooth so that the Taylor approximation (3.4) is sufficiently accurate not to impede convergence.

Finally, let us set some more terminology. The methods fit the general framework of p. 214 of Ortega and Rheinboldt [14]. Our methods are only for the block lower triangular case. But, we were unable to find a naming scheme to reflect that which did not eventually get out of hand when we applied it to the variants given in the next section. Thus, we will base our notation on Ortega and Rheinboldt [14], but we will not use the word "block" in any of the names, since it would have exactly the same obvious meaning in all the names. Notice that for (3.1), Newton's method and the Newton–Gauss–Seidel method are both given by (3.4), and so they are identical. The reason for such attention to names is that things are about to get complicated in the next section.

4. Variants of nonlinear Gauss–Seidel. In this section, we will look at some modifications of the Gauss–Seidel–Newton method given by (3.5). These will consist in using modifications of Newton's method on each block.

First note that for all these variants, besides the obvious advantages of not needing to compute the strict lower triangle of the Jacobian, it would be possible to apply only to the diagonal blocks graph coloring heuristics to compute the derivatives by finite differences or automatic differentiation. And so, based again on the assumption that the F_i can be evaluated independently and assuming also that the cost of $F(x)$ is the same as the sum of the costs of an evaluation of each F_i, one could not do worse than the result of coloring the entire graph. On a parallel machine, there are obvious possibilities for savings by considering coloring, derivative calculations, and factorizations independently for each diagonal block.

4.1. Nonlinear Gauss–Seidel. Steward [15] and Erickson [9] suggest the limiting case of the class of methods we will consider. It is the block version of what Ortega and Rheinboldt [14] call the nonlinear Gauss–Seidel method. In this method, one solves the block systems successively starting from the top left corner and using the newest values as they are found:

$$\text{solve} \quad F_i(x_1^*, \ldots, x_{i-1}^*, x_i) = 0 \quad \text{for } x_i^*, \quad i = 1, 2, \ldots, m. \tag{4.1}$$

This method is defined independently of the particular solution technique applied to the blocks, but Erickson [9] used Newton's method to solve for each x_i^* for his comparisons with Newton's method (3.4). He ran experiments on a 66-variable nonlinear block triangular equation, where the sizes of the 2 diagonal blocks are 44 and 22, respectively.

The ratio he reported of the computing time between Newton's method and the nonlinear Gauss-Seidel method was 2.7. Since Erickson does not specify whether he takes advantage of the block forward substitution approach for solving the linear systems for the Newton step, and since this ratio is very close to 3, which could be estimated by $\sum_{i=1}^m n_i^3 : (\sum_{i=1}^m n_i)^3$ for the savings from doing only the block factorizations rather than the entire matrix, we are not sure how to interpret these results.

4.2. The Gauss–Seidel–Newton method. The Gauss–Seidel principle is to use new information as soon as it is available in order to achieve fast convergence, and this should be especially advantageous for block lower triangular systems.

$$
\begin{aligned}
&x_i^{k+1} = x_i^k - [J_i(x_i^{k,i})]^{-1} F_i(x_i^{k,i}),\\
&\text{where } x_i^{k,i} = (x_1^{k+1}, \ldots, x_{i-1}^{k+1}, x_i^k), \quad J_i(\cdot) = \frac{\partial F_i(\cdot)}{\partial x_i}, \quad i = 1, \ldots, m.
\end{aligned} \tag{4.2}
$$

Below, we will use $x_i^{k,i}$ and J_i without further definition.

It will not surprise the reader that for all the examples we tried, this method converged in less time than Newton's method. There are several reasons for this. First, there is the probability that we will use fewer iterations because of the Gauss–Seidel principle. But even if we use the same number of iterations, we do less function work because only diagonal Jacobian blocks are evaluated rather than the whole matrix. The only savings in linear algebra required for our method compared to Newton's method is that Newton's method has to form the linear combinations of products of blocks in the strict lower triangular part of the Jacobian with already determined components of the Newton step in order to correct the right-hand side of the linear system to be solved for the next component. Test results are given in §5.

4.3. The Gauss–Seidel–Newton stationary (q-step) method. In practice, applying more than one inner iteration to each block can reduce the total number of outer sweeps and the total computing time. Of course, too many inner iterations means that one is doing nonlinear Gauss–Seidel, which has a slower convergence rate than Newton's method.

There is another very interesting issue to consider in deciding how many inner iterations to use. Suppose, for the sake of argument, that $x^0 = (x_1^0, x_2^0)$ has the property that $x_1^0 \neq x_1^*$, but $x_2^0 = x_2^*$. Then, unless one iterates on the first block until convergence, x_1^1 is not x_1^*. Thus, each inner iteration on the second block will be taking us further from x_2^* because it will be trying to converge to a solution to $F_2(x_1^1, x_2) = 0$.

This is an extreme case, but in general one can extend this way of thinking to see that when we begin iterating on a block, it is probably true that the earlier blocks have improved their variables. Therefore, we expect that as we iterate on the ith block toward a solution to $F_i(\bar{x}_{i-1}^{k+1}, x_i) = 0$, initially we will improve our approximation to x_i^*. However, if we keep doing inner iterations, that improvement will end and we will draw further from x_i^* as we continue toward the root of $F_i(x_1^{k+1}, \ldots, x_{i-1}^{k+1}, x_i) = 0$.

The experimental results of several such cases are in §5, including one case in which we diverge if we take more than two inner iterations, but not enough so that we are doing nonlinear Gauss–Seidel.

We found it best to implement the inner iterations in the cheapest way; we use a stationary Newton method in which we do a new block function evaluation at each inner iteration, but only one Jacobian evaluation and factorization per block per outer sweep. The algorithm for q inner stationary Newton steps on each block is as follows:

$$
\begin{aligned}
&x_i^{k(\nu+1)} = x_i^{k(\nu)} - [J_i(x_i^{k,i})]^{-1} F_i(x_i^{k(\nu),i}),\\
&\text{for } \nu = 0, \ldots, q-1, \quad i = 1, \ldots, m, \quad k = 0, 1, 2, \ldots,\\
&\text{and } x_i^{k(\nu),i} = (x_1^{k+1}, \ldots, x_{i-1}^{k+1}, x_i^{k(\nu)}) \quad \text{with } x_i^{k(0)} = x_i^k, \quad x_i^{k(q)} = x_i^{k+1}.
\end{aligned} \tag{4.3}
$$

4.4. The modified Gauss–Seidel–Newton stationary (q-step) method. Another variation, which is well suited to parallel computation, is to evaluate and factor the diagonal blocks concurrently at x^k. We will show in the next section that this modified method and its variations are quadratically convergent. The algorithm for q inner stationary Newton steps on each block is as follows:

$$\begin{aligned} &x_i^{k(\nu+1)} = x_i^{k(\nu)} - [J_i(x^k)]^{-1} F_i(x_i^{k(\nu),i}), \\ &\text{for } \nu = 0, \ldots, q-1, \quad i = 1, \ldots, m, \quad k = 0, 1, 2, \ldots. \end{aligned} \tag{4.4}$$

4.5. Gauss–Seidel–quasi-Newton q-step methods. The more general form for the methods we consider, as well as some others we will consider in future work, is

$$\begin{aligned} &x_i^{k(\nu+1)} = x_i^{k(\nu)} - [B_i^{k(\nu)}]^{-1} F_i(x_i^{k(\nu),i}), \quad B_i^{k(\nu)} \text{ a nonsingular } n_i \times n_i \text{ matrix,} \\ &\quad \text{and } x_i^{k(\nu),i} = (x_1^{k+1}, \ldots, x_{i-1}^{k+1}, x_i^{k(\nu)}) \text{ with } x_i^{k(0)} = x_i^k, \quad x_i^{k(q)} = x_i^{k+1}, \\ &\quad \text{for } \nu = 0, \ldots, q-1, \quad i = 1, \ldots, m, \quad k = 0, 1, 2, \ldots. \end{aligned} \tag{4.5}$$

The notation

$$\bar{x}_i = (x_1, \ldots, x_i) \in R^{n_1} \times R^{n_2} \times \cdots \times R^{n_i}, \qquad i = 1, 2, \ldots, m$$

will be useful in the analysis. Thus, method (4.2) corresponds to (4.5) with

$$B_i^{k(\nu)} = J_i(x_1^{k+1}, \ldots, x_{i-1}^{k+1}, x_i^k), \qquad q = 1, \tag{4.6}$$

method (4.3) is (4.5) with

$$B_i^{k(\nu)} = J_i(x_1^{k+1}, \ldots, x_{i-1}^{k+1}, x_i^k), \tag{4.7}$$

and method (4.4) is (4.5) with

$$B_i^{k(\nu)} = J_i(x_1^k, \ldots, x_i^k). \tag{4.8}$$

4.6. Block Jacobi–Newton. Finally, we include the well-known Jacobi–Newton method for completeness in our test results. It is not a forward nonlinear substitution or Gauss–Seidel method.

$$x_i^{k+1} = x_i^k - [J_i(x_1^k, \ldots, x_i^k)]^{-1} F_i(\bar{x}_i^k), \quad i = 1, \ldots, m, \quad k = 1, \ldots. \tag{4.9}$$

Note that there is no inner iteration; in order to obtain x^{k+1} from x^k, one solves the m independent linear blocks. The same argument is used for all the Jacobian blocks and function blocks.

Besides the simple structure for implementation, the Jacobi method also exhibits a high degree of parallelism. Its main drawback is its slow rate of convergence. We will see this in the analytical convergence results and the experimental results in the following two sections.

5. Convergence results. In this section, we will give some convergence theorems for (4.5). At first, they may seem surprisingly strong, but this is the power of putting the system in block lower triangular form before starting the nonlinear solution process.

Let $|\cdot|$ denote an arbitrary norm defined on each R^{n_i} as well as its subordinate matrix norm. For every i and any $\bar{x}_i \in R^{n_1} \times \cdots \times R^{n_i}$, $i = 1, \ldots, m$, we use the norm

$$\|\bar{x}_i\| = \sum_{j=1}^{i} |x_j|.$$

Remember that $x = \bar{x}_m$, $x^k = \bar{x}_m^k$, $x^* = \bar{x}_m^*$. The Jacobian of F_i with respect to x_i is denoted by $J_i(\bar{x}_i)$.

For all our results we will assume: There exist $x^* \in R^n$, $\epsilon_1 > 0$ such that $F(x)$ is well defined for all x such that $\|x - x^*\| \leq \epsilon_1$, and

$$F(x^*) = 0. \tag{5.1}$$

The Jacobian matrix $J_i(\bar{x}_i)$ exists for all $\bar{x}_i$ such that $\|\bar{x}_i - \bar{x}_i^*\| \leq \epsilon_1$. Moreover, $J_i(\bar{x}_i)$ is continuous at $\bar{x}_i^*$ and $J_i(\bar{x}_i^*)$ is nonsingular for all $i = 1, \ldots, m$. We also assume that for some Γ

$$|J_i(\bar{x}_i)^{-1}| < \Gamma \tag{5.2}$$

for all $i = 1, \ldots, m$ and $\|\bar{x}_i - \bar{x}_i^*\| \leq \epsilon_1$. There exists $L > 0$ such that

$$|J_i(\bar{x}_i) - J_i(\bar{x}_i^*)| \leq L\|\bar{x}_i - \bar{x}_i^*\| \tag{5.3}$$

for all $i = 1, \ldots, m$ and $\|\bar{x}_i - \bar{x}_i^*\| \leq \epsilon_1$. There exists $\alpha \geq 0$ such that

$$|F_i(x_1, \ldots, x_j, \ldots, x_i) - F_i(x_1, \ldots, x_j^*, \ldots, x_i)| \leq \alpha|x_j - x_j^*| \tag{5.4}$$

for all $j \leq i = 1, \ldots, m$ and $\|\bar{x}_i - \bar{x}_i^*\| \leq \epsilon_1$.

LEMMA 5.1.

$$|F_i(\bar{x}_{i-1}, x_i) - F_i(\bar{x}_{i-1}^*, x_i)| \leq \alpha\|\bar{x}_{i-1} - \bar{x}_{i-1}^*\| \tag{5.5}$$

for all $i = 1, \ldots, m$, $\|\bar{x}_i - \bar{x}_i^*\| \leq \epsilon_1$.

Proof. This property can be easily obtained by using (5.4). □

Remark. Observe that we are not assuming differentiability of F_i with respect to variables x_j, $j \neq i$. This allows us to include under our analysis more general functions than the ones considered in the usual convergence analysis for Newton methods.

THEOREM 5.2. *Let* $r \in (0, 1)$. *There exist* ϵ_2 , $\delta > 0$ *such that if*

$$\|x^0 - x^*\| \leq \epsilon_2 \tag{5.6}$$

and

$$|B_i^{k(\nu)} - J_i(\bar{x}_i^*)| \leq \delta \tag{5.7}$$

for all $\nu = 0, \ldots, q-1$, $i = 1, \ldots, m$, $k = 0, 1, 2, \ldots$, *then the sequence* $\{x^k\}$ *generated by* (4.5) *is well defined, converges to* x^*, *and satisfies*

$$|x_i^{k(\nu)} - x_i^*| \leq \frac{r}{m}\|\bar{x}_i^k - \bar{x}_i^*\|, \qquad \nu = 1, \ldots, q, \tag{5.8}$$

$$|x_i^{k+1} - x_i^*| \leq \frac{r}{m}\|\bar{x}_i^k - \bar{x}_i^*\|, \qquad i = 1, \ldots, m, \tag{5.9}$$

and

$$\|x^{k+1} - x^*\| \leq r\|x^k - x^*\|. \tag{5.10}$$

Moreover, for all $\nu = 0, 1, \ldots, q-1$, $i = 1, \ldots, m$ *and* $k = 0, 1, 2, \ldots$, *we have*

(5.11)
$$|x_i^{k(\nu+1)} - x_i^*| \leq \Gamma(|B_i^{k(\nu)} - J_i(\bar{x}_i^*)| + L|x_i^{k(\nu)} - x_i^*|)|x_i^{k(\nu)} - x_i^*| + \Gamma\alpha\|\bar{x}_{i-1}^{k+1} - \bar{x}_{i-1}^*\|.$$

Proof. See the Appendix.

The following result applies to the Gauss–Seidel–Newton (4.3) and modified Gauss–Seidel–Newton methods (4.4) with q inner iterations.

THEOREM 5.3. *Assume that the choice of* $B_i^{k(\nu)}$ *in* (4.5) *is given by* (4.7) (*the Gauss–Seidel–Newton*) *or by* (4.8) (*the modified Gauss–Seidel–Newton*). *Then, there exists* $\epsilon_3 > 0$ *such that if*

$$\|x^0 - x^*\| \leq \epsilon_3,$$

the sequence generated by (4.5) *is well defined and converges to* x^*. *Furthermore, there exists* c_1 *such that*

$$|x_i^{k(\nu)} - x_i^*| \leq c_1\|\bar{x}_i^k - \bar{x}_i^*\|^{\nu+1} \tag{5.12}$$

so that

$$|x_i^{k+1} - x_i^*| \leq c_1\|\bar{x}_i^k - \bar{x}_i^*\|^{q+1} \tag{5.13}$$

and

$$\|x^{k+1} - x^*\| \leq c_1\|x^k - x^*\|^{q+1}, \tag{5.14}$$

where $\nu = 0, \ldots, q$, $i = 1, \ldots, m$, *and* $k = 0, 1, 2, \ldots$.

Proof. See the Appendix.

The rest of this section is dedicated to an analysis of the Jacobi–Newton method discussed in §3. Recall that an iteration of this method is defined by

$$x_i^{k+1} = x_i^k - J_i(\bar{x}_i^k)^{-1}F_i(\bar{x}_i{}^k) \tag{5.15}$$

for $i = 1, \ldots, m$. It turns out that in a sense, the method is m-step quadratically convergent.

THEOREM 5.4. There exist $\epsilon_4, c_2 > 0$ such that, if

$$\|x^0 - x^*\| \leq \epsilon_4, \tag{5.16}$$

the sequence generated by the Jacobi–Newton method (5.15) is well defined, converges to x^*, and satisfies

$$|x_i^{k+1} - x_i^*| \leq c_2(|x_i^k - x_i^*|^2 + \cdots + |x_1^{k-i+1} - x_1^*|^2) \tag{5.17}$$

for all $i = 1, \ldots, m$ and $k \geq m - 1$.

Moreover, defining

$$y^k = (x_1^k, x_2^{k+1}, \ldots, x_m^{k+m-1}),$$

there exists a constant $c_3 > 0$ such that

$$\|y^{k+1} - x^*\| \leq c_3\|y^k - x^*\|^2 \tag{5.18}$$

for all $k = 0, 1, 2, \ldots$.

Proof. See the Appendix.

6. Experimental results. We generated our test problems from the Moré–Garbow–Hillstrom [13] test set. Some are polynomial and some are combinations of polynomial and trigonometric functions. Such simple functions would not favor the methods studied here because they are so cheap to differentiate. Thus, Newton's method does not have to pay much of a penalty for the extraneous derivative calculations.

In all cases, the diagonal blocks were fairly large, which again means that Newton's method does not have to do as much extraneous differentiation in computing the strict lower triangular part of the Jacobian as if the diagonal blocks were small. But the diagonal blocks were the same size, which our intuition says might favor a Gauss–Seidel method because it would maximize the average amount of new information available to each block iteration. However, all diagonal blocks of the same size would minimize the extra linear algebra needed by Newton's method for correcting the right-hand sides in computing the Newton step.

The computer we used is a single processor of Intel 80386 in the Sequent Symmetry system.

We wanted to avoid having to code Tarjan's algorithm [16] just to test our ideas, and so we used systems already in block triangular form with known solutions.

The test problems we used were generated in a simple way from some variable-dimension problems in Moré, Garbow, and Hillstrom [13]. Given any block dimension n and number of blocks m, we generated some polynomial test functions from two variable-dimension test problems as follows: Let the function $F_a : R^n \to R^n$ be the block of nonlinear equations with n unknowns $y = (y_1, y_2, \ldots, y_n)$:

$$f_i(y) = y_i + \sum_{j=1}^{n} y_j - (n+1), \qquad 1 \le i \le n-1,$$

$$f_n(y) = \left(\prod_{j=1}^{n} y_j \right) - 1.$$

Let $F_b : R^n \to R^n$ be another block of nonlinear equations with n unknowns:

$$f_i(y) = (3 - 2y_i)y_i - y_{i-1} - 2y_{i+1} + 1, \qquad i = 1, \ldots, n.$$

Now generate an $n \cdot m \times n \cdot m$ system in the blocks of variables $x = (x_1, x_2, \ldots, x_m)$ by

$$F(x) = \begin{cases} F_1(x_1) = F_a(x_1), \\ F_2(x_1, x_2) = F_a(x_1) + F_b(x_2), \\ \vdots \\ F_{2\ell}(x_1, x_2, \ldots, x_{2\ell}) = F_a(x_1) + \prod_{j=1}^{\ell-1} F_b(x_{2j}) F_a(x_{2j+1}) + F_b(x_{2\ell}), \\ F_{2\ell+1}(x_1, x_2, \ldots, x_{2\ell+1}) = F_a(x_1) + \left[\prod_{j=1}^{\ell-1} F_b(x_{2j}) F_a(x_{2j+1}) \right] F_b(x_{2\ell}) \\ \qquad\qquad + F_a(x_{2\ell+1}), \\ \vdots \\ F_m(x_1, x_2, \ldots, x_m). \end{cases}$$

The second group of functions were generated from F_a, F_b, and the trigonometric system F_c with n unknowns:

$$f_i(y) = n - \sum_{j=1}^{n} \cos\ y_j + i(1 - \cos\ y_i) - \sin\ y_i, \qquad i = 1, \ldots, n$$

as follows:

$$F(x) = \left\{ \begin{array}{l} F_1(x_1) = F_a(x_1), \\ F_2(x_1, x_2) = F_a(x_1) + F_b(x_2), \\ F_3(x_1, x_2, x_3) = F_a(x_1) + F_b(x_2) + F_c(x_3), \\ \vdots \\ F_{3\ell+1}(x_1, \ldots, x_{3\ell+1}) \\ \quad = F_a(x_1) + [\prod_{j=1}^{\ell-1} F_b(x_{3j-1}) F_c(x_{3j}) F_a(x_{3j+1})] F_b(x_{3\ell-1}) F_c(x_{3\ell}) \\ \qquad + F_a(x_{3\ell+1}) \\ F_{3\ell+2}(x_1, \ldots, x_{3\ell+2}) \\ \quad = F_a(x_1) + \prod_{j=1}^{\ell} F_b(x_{3j-1}) F_c(x_{3j}) F_a(x_{3j+1}) + F_b(x_{3\ell+2}) \\ F_{3\ell+3}(x_1, \ldots, x_{3\ell+3}) \\ \quad = F_a(x_1) + [\prod_{j=1}^{\ell} F_b(x_{3j-1}) F_c(x_{3j}) F_a(x_{3j+1})] F_b(x_{3\ell+2}) \\ \qquad + F_c(x_{3\ell+3}) \\ \vdots \\ F_m(x_1, x_2, \ldots, x_m) \end{array} \right\} = 0.$$

Here we report results on a single processor to demonstrate that the methods given here are very good sequential algorithms. In later work, we will test parallel variants of the Gauss–Seidel–Newton approach for systems of the form (3.1).

In all cases, to keep things simple the starting value of x^0 was reasonably close to the solution, and no globalizing strategy was used. The stopping criterion was to reduce the total function ℓ_2 norm below 10^{-12}.

First, we compare the performance of the five methods on a 600×600 polynomial block triangular system with six blocks of 100 variables (see Fig. 1).

As expected, the Jacobi–Newton method took the most time to converge. The nonlinear Gauss–Seidel method took 13 iterations for each block, and total computing time was impressively small. The Gauss–Seidel–Newton method used one iteration less than the Newton's method, but the computing time was about 27% because of less time spent in Jacobian evaluation. We do not compute the strict lower triangle, and in doing linear algebra, Newton must correct the right-hand sides in solving for the Newton step.

The Gauss–Seidel–Newton method with more than one inner iteration was the most effective one among all the methods for this problem. The experimental results show that the number of outer iterations is sharply decreased from 13 to 5 when the number of inner iterations is increased to 2. However, the number of outer iterations decreases more slowly as the inner number of iterations increases further. The optimal number of stationary inner iterations is problem dependent. Our experiments show that when $q = 4$, then $k = 3$ was needed, and this minimized the computing time for this system.

Next we tested a 1600×1600 polynomial block triangular problem with 16 blocks of 100 variables (see Fig. 2). Unfortunately, the Jacobi–Newton method could not converge to a solution from the same x^0 as the other methods used. We modified the initial value so that it was closer to a solution, and we found the solution by the Jacobi–Newton method. This group of experiments also showed that the nonlinear Gauss–Seidel method and the Gauss–Seidel–Newton with stationary inner iterations converged faster than the Newton's method.

We also tested the five methods on the second group of nonlinear block triangular problems with combined polynomial and trigonometric functions with 8 blocks of 100

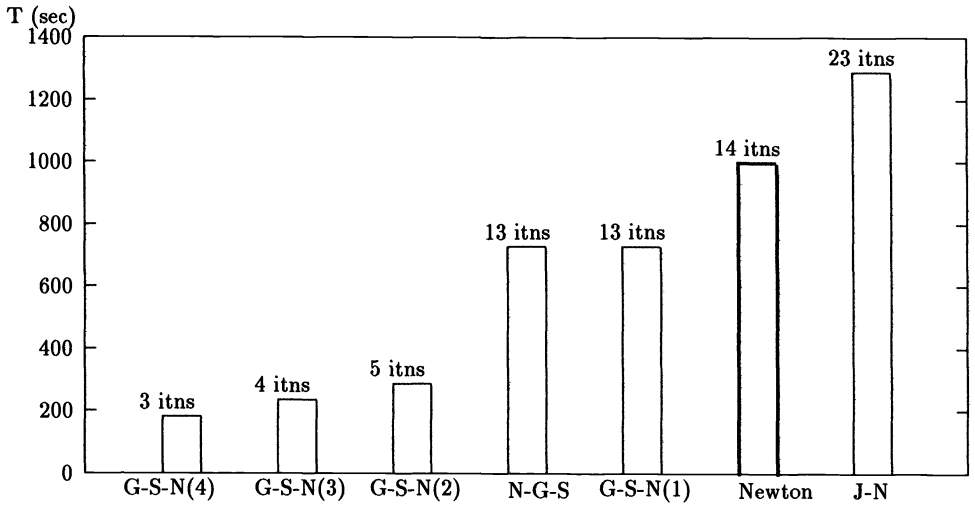


FIG. 1. *One-processor times for different methods on a* 600 × 600 *polynomial system.*

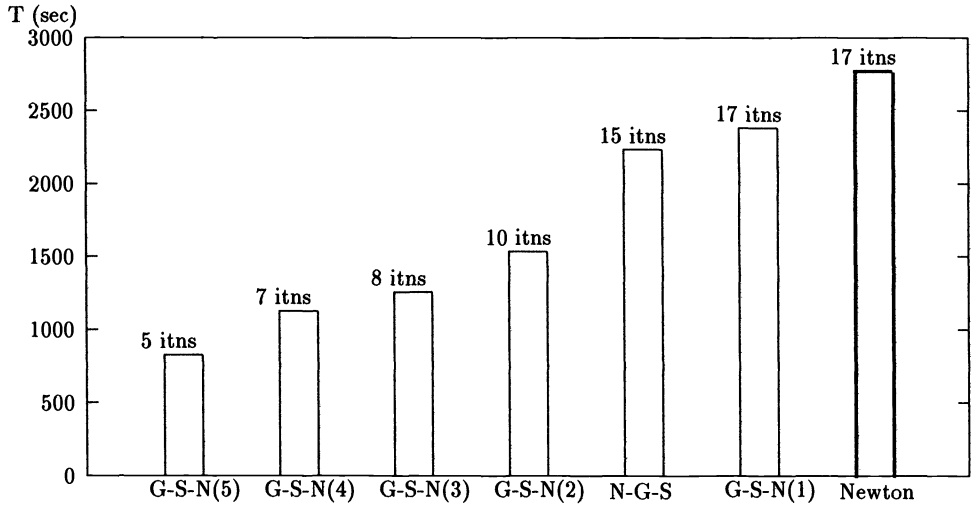


FIG. 2. *One-processor times for different methods on a* 1600 × 1600 *polynomial system.*

variables and 16 blocks of 100 variables, respectively (see Figs. 3 and 4). Performance was similar to the first tests, but the Jacobi–Newton method again failed to converge on the larger problem of the second group. Also, the Gauss–Seidel–Newton method was not as effective as before. The largest number of inner iterations successfully applied was 2, except for the nonlinear Gauss–Seidel method which in a sense has infinitely many inner iterations. When more inner iterations than 2 were applied, the method did not converge to the solution (see Figs. 3 and 4).

In general, more inner iterations should make the Gauss–Seidel–Newton method more likely to converge. But, as we discussed in §3, this is not always true. The experiments presented in Figs. 3 and 4 are examples of this point.

The best way to handle this situation is to adaptively choose the number of inner iterations for each block by some easily imagined strategies, such as reaching a target percentage reduction in the block function norms. It is not our purpose in this paper

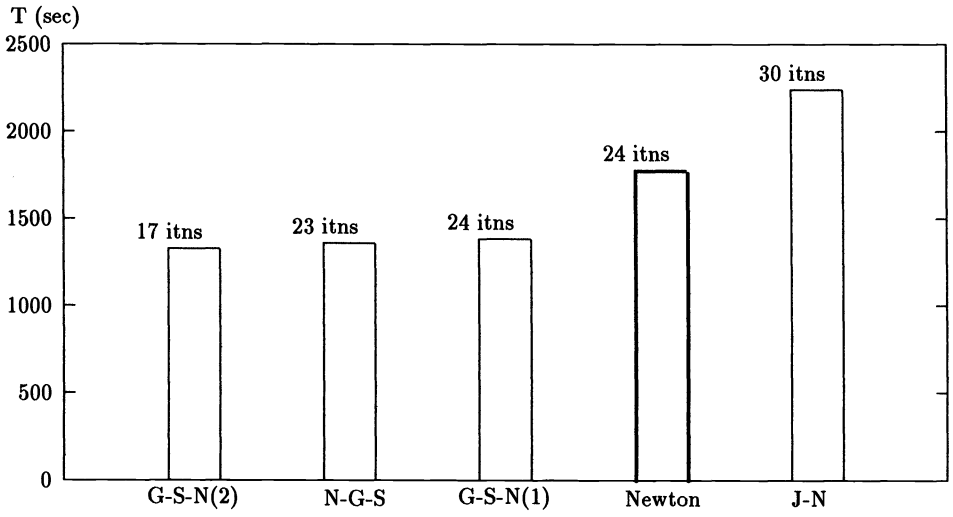


FIG. 3. *One-processor times for different methods on a* 800 × 800 *polynomial, trigonometric coupled variable function.*

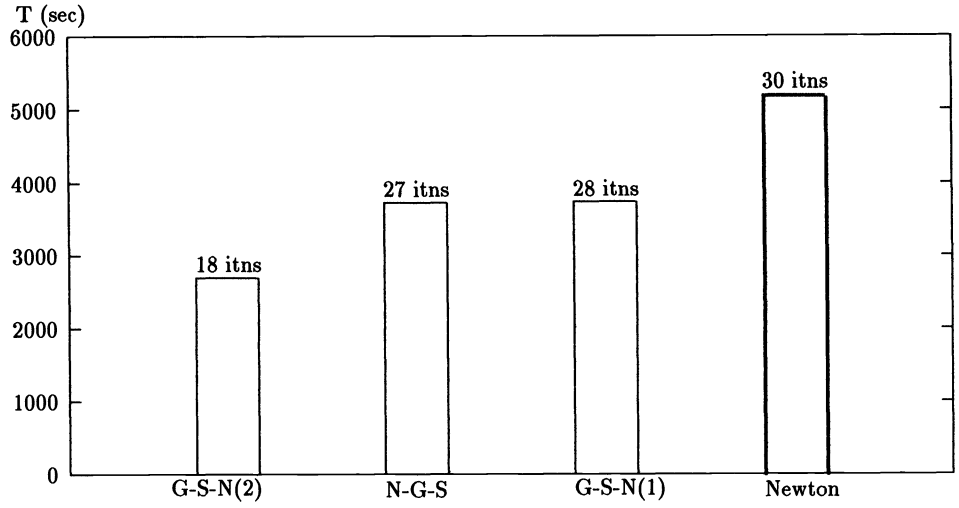


FIG. 4. *One-processor times for different methods on a* 1600 × 1600 *polynomial, trigonometric coupled variable function.*

to complicate the basic idea with such implementational details, however important to a production version they might be. But to illustrate the point, we did some experiments to show how convergence can vary with the choice of q_i.

For example, the best result we got for the 800 × 800 problem with eight blocks of 100 variables was to use in the first and second outer iterations, respectively, 1, 2, 3, 2, 2, 2, 3, 3 inner iterations. In the third outer iteration, 2 inner iterations were applied to the first block, and 3 to the other blocks. We obtained convergence with five more outer iterations, with 3 inner iterations applied to each of the blocks.

This sums to 8 total outer iterations and 1024 seconds spent, which reduced the computing time approximately 25% over the solution by the Gauss–Seidel–Newton method with the best uniform number of inner iterations. Similarly, for the 1600 × 1600 problem with 16 blocks of 100 variables, 10 outer iterations and 1995 seconds

were used with varying numbers of inner iterations, which was approximately a 30% reduction in computing time over the best performance by applying a fixed number of inner iterations to each block.

7. Summary and future research. We have shown that Gauss–Seidel–Newton algorithms can be much more effective than Newton's method applied to nonlinear systems put into block triangular form by standard graph algorithms used to permute general sparse linear systems to block lower triangular form.

We have introduced the notion of allowing stationary inner iterations in the Gauss–Seidel–Newton method, and we have seen that the Jacobi–Newton method, which has the highest degree of parallelism among the methods, has the slowest convergence rate, and seems less effective.

The development of secant approximations to the diagonal blocks of a block triangular Jacobian is an attractive research topic, since it would allow cheap approximations and it would allow the factorization of the Jacobian approximation to be updated efficiently. We have done some analytical studies on quasi-Newton methods in which the block diagonals are approximated by different updates such as the Broyden update (see Broyden [1]). We plan to implement these methods on parallel computers and to test the performance.

The algorithms we considered here all have various synchronization points in them. For Jacobi and Gauss–Seidel methods, a potentially attractive alternative is to allow parallel processors to proceed asynchronously. We will develop synchronous and asynchronous parallel methods based on the sequential methods we have studied in this paper.

A. Proofs of the convergence results.

LEMMA A1. *Suppose that*

$$\{e_i^\nu\,,\ \ i=1,\dots,m,\ \ \nu=0,1,\dots,q\} \tag{A.1}$$

is a set of positive real numbers such that

$$\sum_{i=1}^{m} e_i^0 \le \epsilon, \tag{A.2}$$

$$e_i^{\nu+1} \le \Gamma(\delta + Le_i^\nu)e_i^\nu + \Gamma\alpha\sum_{j=1}^{i-1} e_j^q \tag{A.3}$$

for $i=1,\dots,m$ *and* $\nu=0,1,\dots,q-1$. *By convention* $\sum_{j=1}^{0}=0$.

Then, there exist continuous functions $\phi_{i,\nu}(\epsilon,\delta)$, $i=1,\dots,m$, $\nu=1,\dots,q$ *such that*

$$\phi_{i,\nu}(0,0)=0 \tag{A.4}$$

and

$$e_i^\nu \le \phi_{i,\nu}(\epsilon,\delta)\sum_{j=1}^{i} e_j^0 \tag{A.5}$$

for all $i=1,\dots,m$ *and* $\nu=1,\dots,q$.

Proof. We proceed by induction on i. Let us first prove (A.5) for $i = 1$ by induction on ν. For $\nu = 1$, we have by (A.2) and (A.3) that

$$e_1^\nu = e_1^1 \leq \Gamma(\delta + Le_1^0)e_1^0 \leq \Gamma(\delta + L\epsilon)e_1^0 .$$

Therefore, (A.5) holds with

$$\phi_{1,1}(\epsilon, \delta) = \Gamma(\delta + L\epsilon) . \tag{A.6}$$

Now let us prove (A.4) and (A.5) for $i = 1$ and $\nu \in \{1, \ldots, q\}$. The inductive hypothesis is that $\phi_{1,\ell}$ is defined and satisfies

$$\phi_{1,\ell}(0, 0) = 0$$

and

$$e_1^\ell \leq \phi_{1,\ell}(\epsilon, \delta)e_1^0$$

for $\ell = 1, \ldots, \nu - 1$.

By (A.3) and the inductive hypothesis, we have that

$$\begin{aligned} e_1^\nu &\leq \Gamma(\delta + Le_1^{\nu-1})e_1^{\nu-1} \\ &\leq \Gamma\left(\delta + L\phi_{1,\nu-1}(\epsilon, \delta)e_1^0\right) \phi_{i,\nu-1}(\epsilon, \delta)e_1^0 \\ &\leq \left[\Gamma\left(\delta + L\phi_{1,\nu-1}(\epsilon, \delta)\epsilon\right)\right] \phi_{1,\nu-1}(\epsilon, \delta)e_1^0. \end{aligned}$$

Thus (A.5) holds for $i = 1$ defining

$$\phi_{1,\nu}(\epsilon, \delta) = \Gamma(\delta + L\phi_{1,\nu-1}(\epsilon, \delta)\epsilon)\phi_{1,\nu-1}(\epsilon, \delta).$$

By the inductive hypothesis, $\phi_{1,\nu}$ is continuous and $\phi_{1,\nu}(0, 0) = 0$. Therefore, (A.5) is proved for $i = 1, \nu = 1, \ldots, q$.

Assume, by inductive hypothesis, that $\phi_{j,\ell}$ is defined and satisfies

$$\phi_{j,\ell}(0, 0) = 0$$

and

$$e_j^\ell \leq \phi_{j,\ell}(\epsilon, \delta) \sum_{s=1}^{j} e_s^0$$

for all $j \in \{1, \ldots, i-1\}$, $\ell = 1, \ldots, q$.

Let us prove (A.4) and (A.5) for a given $i > 1$, $\nu = 1, \ldots, q$. We proceed by induction on ν. If $\nu = 1$, we have, by (A.3) and the inductive hypothesis, that

$$\begin{aligned} e_i^1 &\leq \Gamma(\delta + Le_i^0)e_i^0 + \Gamma\alpha \sum_{j=1}^{i-1} e_j^q \\ &\leq \Gamma(\delta + L\epsilon)e_i^0 + \Gamma\alpha \sum_{j=1}^{i-1} \phi_{j,q}(\epsilon, \delta) \sum_{\ell=1}^{j} e_\ell^0 \\ &\leq \Gamma(\delta + L\epsilon) \sum_{j=1}^{i} e_j^0 + \Gamma\alpha \left(\sum_{j=1}^{i-1} \phi_{j,q}(\epsilon, \delta) \right) \left(\sum_{\ell=1}^{i-1} e_\ell^0 \right) \\ &\leq \left[\Gamma(\delta + L\epsilon) + \Gamma\alpha \sum_{j=1}^{i-1} \phi_{j,q}(\epsilon, \delta) \right] \sum_{\ell=1}^{i} e_\ell^0. \end{aligned}$$

Thus (A.5) holds with

$$\phi_{i,1}(\epsilon,\delta) = \Gamma(\delta + L\epsilon) + \Gamma\alpha \sum_{j=1}^{i-1} \phi_{j,q}(\epsilon,\delta)\ .$$

If $\nu > 1$, we have by (A.3) and the inductive hypothesis that

$$\begin{aligned}
e_i^\nu &\le \Gamma(\delta + Le_i^{\nu-1})e_i^{\nu-1} + \Gamma\alpha \sum_{j=1}^{i-1} e_j^q \\
&\le \Gamma\Big(\delta + L\phi_{i,\nu-1}(\epsilon,\delta)\sum_{j=1}^{i} e_j^0\Big)\phi_{i,\nu-1}(\epsilon,\delta)\sum_{j=1}^{i} e_j^0 \\
&\quad + \Gamma\alpha \sum_{j=1}^{i-1} \phi_{j,q}(\epsilon,\delta)\sum_{\ell=1}^{j} e_\ell^0 \\
&\le \Gamma\left(\delta + L\phi_{i,\nu-1}(\epsilon,\delta)\epsilon\right)\phi_{i,\nu-1}(\epsilon,\delta)\Big(\sum_{j=1}^{i} e_j^0\Big) \\
&\quad + \Gamma\alpha\Big(\sum_{j=1}^{i-1} \phi_{j,q}(\epsilon,\delta)\Big)\Big(\sum_{j=1}^{i} e_j^0\Big) \\
&\le \left[\Gamma\left(\delta + L\phi_{i,\nu-1}(\epsilon,\delta)\epsilon\right)\phi_{i,\nu-1}(\epsilon,\delta) + \Gamma\alpha\Big(\sum_{j=1}^{i-1} \phi_{j,q}(\epsilon,\delta)\Big)\right]\sum_{j=1}^{i} e_j^0.
\end{aligned}$$

Thus (A.5) holds, defining

$$\phi_{i,\nu}(\epsilon,\delta) = \Gamma(\delta + L\phi_{i,\nu-1}(\epsilon,\delta)\epsilon)\phi_{i,\nu-1}(\epsilon,\delta) + \Gamma\alpha \sum_{j=1}^{i-1} \phi_{j,q}(\epsilon,\delta)\ .$$

By the inductive hypothesis, $\phi_{i,\nu}$ is continuous and $\phi_{i,\nu}(0,0) = 0$. Therefore, the existence of the functions $\phi_{i,\nu}$ such that (A.4) and (A.5) hold is proved. □

Proof of Theorem 5.2. By (4.2), $|J_i(\bar{x}_i^*)^{-1}| < \Gamma$. Therefore, we can define $\delta_1 > 0$ so that $|B_i^{-1}| \le \Gamma$ for all $B_i \in \mathbb{R}^{n_i \times n_i}$ such that $|B_i - J_i(\bar{x}_i^*)| \le \delta_1$ for $i = 1, \ldots, m$.

Let ϵ_2, δ be such that $0 < \epsilon_2 \le \epsilon_1$, $0 < \delta \le \delta_1$, and

$$\phi_{i,\nu}(\epsilon_2,\delta) \le \frac{r}{m} \tag{A.7}$$

for all $i = 1, \ldots, m$ and $\nu = 1, \ldots, q$, where the functions $\phi_{i,\nu}$ are given by Lemma A1.

Assume (4.6) and (4.7). Let us prove inductively that

$$\|(x_i^{0(\nu),i}) - \bar{x}_i^*\| \le \epsilon_2 \tag{A.8}$$

for all $i = 1, \ldots, m$, $\nu = 0, 1, \ldots, q$ and that

$$|x_i^{0(\nu)} - x_i^*| \le \frac{r}{m}\sum_{j=1}^{i} |x_j^0 - x_j^*| \tag{A.9}$$

for all $i = 1, \ldots, m$, $\nu = 1, \ldots, q$.

We proceed by induction on i. So, let us first prove (A.8) and (A.9) for $i = 1$. If $i = 1$ and $\nu = 0$, (A.8) is trivial. If $i = 1$ and $\nu = 1$, we have, by (4.3), (A.1), (A.6), (A.7), and (4.6), that

$$\begin{aligned}
|x_1^{0(1)} - x_1^*| &= |x_1^0 - x_1^* - (B_1^{0(0)})^{-1} F_1(x_1^0)| \\
&= \left| x_1^0 - x_1^* - (B_1^{0(0)})^{-1} \left(\int_0^1 J_1\left(x_1^* + t(x_1^0 - x_1^*)\right) dt \right) (x_1^0 - x_1^*) \right| \\
&= \left| I - (B_1^{0(0)})^{-1} J_1(x_1^*) \right| |x_1^0 - x_1^*| + \Gamma L |x_1^0 - x_1^*|^2 \\
&\leq \Gamma\delta |x_1^0 - x_1^*| + \Gamma L |x_1^0 - x_1^*|^2 \\
&\leq \Gamma(\delta + L\epsilon_2)|x_1^0 - x_1^*| = \phi_{1,1}(\delta, \epsilon_2)|x_1^0 - x_1^*| \\
&\leq \frac{r}{m}|x_1^0 - x_1^*| \leq \epsilon_2 .
\end{aligned}$$

So, (A.8) and (A.9) are proved for $i = 1$, $\nu = 1$.

Assuming that (A.8) and (A.9) are true for $i = 1$ and $\nu \in \{1, \ldots, q-1\}$, let us prove these two inequalities for $i = 1$, replacing ν by $\nu + 1$.

By (4.3), (A.1), (A.6), (A.7), (4.6), (4.5), and the inductive hypothesis, we have that

$$\begin{aligned}
|x_1^{0(\nu+1)} - x_1^*| &= |x_1^{0(\nu)} - x_1^* - (B_1^{0(\nu)})^{-1} F_1(x_1^{0(\nu)})| \\
&\leq \left| x_1^{0(\nu)} - x_1^* - (B_1^{0(\nu)})^{-1} \left(\int_0^1 J_1(x_1^* + t(x_1^{0(\nu)} - x_1^*)) dt \right) (x_1^{0(\nu)} - x_1^*) \right| \\
&\leq |I - (B_1^{0(\nu)})^{-1} J_1(x_1^*)| \, |x_1^{0(\nu)} - x_1^*| \\
&\quad + \Gamma L |x_1^{0(\nu)} - x_1^*|^2 \\
&\leq \Gamma \left(|B_1^{0(\nu)} - J_1(x_1^*)| + L|x_1^{0(\nu)} - x_1^*| \right) |x_1^{0(\nu)} - x_1^*| \\
&\leq \Gamma(\delta + L|x_1^{0(\nu)} - x_1^*|)|x_1^{0(\nu)} - x_1^*|
\end{aligned}$$

So, by (A.3), (A.5), (4.6), and (A.7),

$$|x_1^{0(\nu+1)} - x_1^*| \leq \phi_{1,\nu+1}(\epsilon_2, \delta)|x_1^0 - x_1^*| \leq \frac{r}{m}|x_1^0 - x_1^*| .$$

Therefore, (A.8) and (A.9) are proved for $i = 1$ and $\nu = 1, \ldots, q$.

Let us prove now that (A.8) and (A.9) hold for an arbitrary $i \in \{2, \ldots, m\}$. Assume, as the inductive hypothesis, that

$$\|(x_j^{0(\nu),j}) - \bar{x}_j^*\| \leq \epsilon_2$$

for all $j = 1, \ldots, i-1$, $\nu = 0, 1, \ldots, q$ and that

$$|x_j^{0(\nu)} - x_j^*| \leq \frac{r}{m} \sum_{\ell=1}^{j} |x_\ell^0 - x_\ell^*|$$

for all $j = 1, \ldots, i-1$, $\nu = 1, \ldots, q$.

We first prove (A.8) for $\nu = 0$. By the inductive hypothesis and (4.6), we have that

$$\|x_i^{0,i} - \bar{x}_i^*\| = \|(\bar{x}_{i-1}^1, x_i^0) - (\bar{x}_{i-1}^*, x_i^*)\|$$

$$
\begin{aligned}
&= \|\bar{x}_{i-1}^1 - \bar{x}_{i-1}^*\| + |x_i^0 - x_i^*| \\
&= \sum_{j=1}^{i-1} |x_j^1 - x_j^*| + |x_i^0 - x_i^*| \\
&\le \sum_{j=1}^{i-1} \frac{r}{m} \sum_{\ell=1}^{j} |x_\ell^0 - x_\ell^*| + |x_i^0 - x_i^*| \\
&\le \sum_{j=1}^{i-1} \frac{r}{m} \sum_{\ell=1}^{i-1} |x_\ell^0 - x_\ell^*| + |x_i^0 - x_i^*| \\
&\le \sum_{\ell=1}^{i-1} |x_\ell^0 - x_\ell^*| \sum_{j=1}^{i-1} \frac{r}{m} + |x_i^0 - x_i^*| \\
&\le \sum_{\ell=1}^{i-1} |x_\ell^0 - x_\ell^*| + |x_i^0 - x_i^*| = \|\bar{x}_i^0 - \bar{x}_i^*\| \le \epsilon_2 ,
\end{aligned}
$$

since $i \le m$ and $r < 1$.

Now let us prove (A.8) and (A.9), replacing ν by $\nu + 1$. The proof of (A.8) and (A.9) for $\nu = 1$ is the same as the following proof, with the obvious substitutions.

By (4.3), (A.1), (A.6), (A.7), (4.6), (4.5), and the inductive hypothesis, we have that

(A.10)

$$
\begin{aligned}
|x_i^{0(\nu+1)} - x_i^*| &= |x_i^{0(\nu)} - x_i^* - (B_i^{0(\nu)})^{-1} F_i(x_i^{0(\nu),i})| \\
&\le |x_i^{0,\nu} - x_i^* - (B_i^{0(\nu)})^{-1} F_i(\bar{x}_{i-1}^*, x_i^{0(\nu)})| \\
&\quad + \Gamma |F_i(\bar{x}_{i-1}^1, x_i^{0(\nu)}) - F_i(\bar{x}_{i-1}^*, x_i^{0(\nu)})| \\
&\le \left| x_i^{0(\nu)} - x_i^* - (B_i^{0(\nu)})^{-1} \left(\int_0^1 J_i(\bar{x}_{i-1}^*, x_i^* + t(x_i^{0(\nu)} - x_i^*)) dt \right) \right. \\
&\quad \left. (x_i^{0(\nu)} - x_i^*) \right| + \Gamma\alpha \|\bar{x}_{i-1}^1 - \bar{x}_{i-1}^*\| \\
&\le |I - (B_i^{0(\nu)})^{-1} J_i(\bar{x}_i^*)| \, |x_i^{0(\nu)} - x_i^*| \\
&\quad + \Gamma L |x_i^{0(\nu)} - x_i^*|^2 + \Gamma\alpha \|\bar{x}_{i-1}^1 - \bar{x}_{i-1}^*\| \\
&\le \Gamma \left(|B_i^{0(\nu)} - J_i(\bar{x}_i^*)| + L |x_i^{0(\nu)} - x_i^*| \right) |x_i^{0(\nu)} - x_i^*| \\
&\quad + \Gamma\alpha \|\bar{x}_{i-1}^1 - \bar{x}_{i-1}^*\| \\
&\le \Gamma(\delta + L |x_i^{0(\nu)} - x_i^*|) |x_i^{0(\nu)} - x_i^*| + \Gamma\alpha \sum_{j=1}^{i-1} |x_j^1 - x_j^*| .
\end{aligned}
$$

So, by (A.3), (A.5), (4.6), (A.7), and the inductive hypothesis,

$$
|x_i^{0(\nu+1)} - x_i^*| \le \phi_{i,\nu+1}(\epsilon_2, \delta) \sum_{j=1}^{i} |x_j^0 - x_j^*| \le \frac{r}{m} \sum_{j=1}^{i} |x_j^0 - x_j^*| . \tag{A.11}
$$

But, by the inductive hypothesis,

$$|x_j^1 - x_j^*| \le \frac{r}{m}\sum_{\ell=1}^{j}|x_\ell^0 - x_\ell^*| \le \frac{r}{m}\sum_{\ell=1}^{i}|x_\ell^0 - x_\ell^*| \tag{A.12}$$

for $j = 1, \dots, i-1$.

Adding the inequalities (A.11) and (A.12), we obtain

$$\begin{aligned}\sum_{j=1}^{i-1}|x_j^1 - x_j^*| + |x_i^{0(\nu+1)} - x_i^*| &\le \frac{(i-1)r}{m}\sum_{j=1}^{i}|x_j^0 - x_j^*| \\ &+ \frac{r}{m}\sum_{j=1}^{i}|x_j^0 - x_j^*| = \frac{ir}{m}\|\bar{x}_i^0 - \bar{x}_i^*\| \le \epsilon_2.\end{aligned} \tag{A.13}$$

With (A.11) and (A.13), the inductive proof of (A.8) and (A.9) is complete. Therefore,

$$|x_i^1 - x_i^*| \le \frac{r}{m}\sum_{j=1}^{m}|x_j^0 - x_j^*| = \frac{r}{m}\|x^0 - x^*\|$$

for all $i = 1, \dots, m$.

So,

$$\|x^1 - x^*\| \le r\|x^0 - x^*\|.$$

Repeating the argument used for $k = 0$ we prove that (4.8)–(4.11) holds for $k = 0, 1, 2, \dots$. This completes the proof of the theorem. □

Proof of Theorem 5.3. Let ϵ_2 and δ be as given by Theorem 5.2 for $r = \frac{1}{2}$. Let $\epsilon_3 \le \epsilon_2$ be such that

$$|J_i(\bar{x}_i) - J_i(\bar{x}_i^*)| \le \delta$$

for all $\bar{x}_i$ such that $\|\bar{x}_i - \bar{x}_i^*\| \le \epsilon_3$. With this choice, the convergence of the sequences generated by the Gauss–Seidel–Newton method and the modified Gauss–Seidel–Newton method follows from Theorem 5.2.

Let us prove by induction that there exist constants $c_{i,\nu}$ that only depend on Γ, L, ϵ_3, and α such that

$$|x_i^{k(\nu)} - x_i^*| \le c_{i,\nu}\|\bar{x}_i^k - \bar{x}_i^*\|^{\nu+1} \tag{A.14}$$

for $i = 1, \dots, m$, $\nu = 0, 1, 2, \dots, q$.

First, we prove (A.14) for the choice (3.7) of $B_i^{k(\nu)}$. We proceed by induction on i. For $i = 1$, we prove (A.14) by induction on ν.

For $\nu = 0$, (A.14) is trivial with $c_{1,\nu} = 1$. Let us make the inductive step on ν. By (5.11) and (4.6) we have

$$\begin{aligned}|x_1^{k(\nu+1)} - x_1^*| \le \Gamma(&|J_1(x_1^{k,1}) - J_1(x_1^*)| \\ &+ L|x_1^{k(\nu)} - x_1^*|)|x_1^{k(\nu)} - x_1^*|.\end{aligned}$$

So, by (5.3),

$$\begin{aligned}|x_1^{k(\nu+1)} - x_1^*| &\le \Gamma L(|x_1^k - x_1^*| + |x_1^{k(\nu)} - x_1^*|)|x_1^{k(\nu)} - x_1^*| \\ &\le \Gamma L(|x_1^k - x_1^*| + c_{1,\nu}\|x_1^k - x_1^*\|^{\nu+1})c_{1,\nu}\|x_1^k - x_1^*\|^{\nu+1} \\ &\le \Gamma L(|x_1^k - x_1^*| + c_{1,\nu}\|x_1^k - x_1^*\|^{\nu+1})c_{1,\nu}\|x_1^k - x_1^*\|^{\nu+1} \\ &\le \Gamma L(1 + c_{1,\nu}\|x_1^k - x_1^*\|^{\nu})c_{1,\nu}\|x_1^k - x_1^*\|^{\nu+2}.\end{aligned}$$

Therefore, the inductive step is complete, defining

$$c_{1,\nu+1} = \Gamma L(1 + c_{1,\nu}\epsilon_3^{\nu})c_{1,\nu}.$$

Assume now that $|x_j^{k(\nu)} - x_j^*| \le c_{j,\nu}\|\bar{x}_j^k - \bar{x}_j^*\|^{\nu+1}$ for all $j \in \{1, \ldots, i-1\}, \nu \in \{0, 1, 2, \ldots, q\}$. Let us prove that (A.14) also holds for i. We proceed by induction on ν. For $\nu = 0$, (A.14) is trivial with $c_{i,\nu} = 1$. Let us make the inductive step. By (5.11) and (4.6) we have

$$\begin{aligned}|x_i^{k(\nu+1)} - x_i^*| \le \Gamma(&|J_i(x_i^{k,i}) - J_i(\bar{x}_i^*)| \\ &+ L|x_i^{k(\nu)} - x_i^*|)|x_i^{k(\nu)} - x_i^*| + \Gamma\alpha\|\bar{x}_{i-1}^{k+1} - \bar{x}_{i-1}^*\|.\end{aligned}$$

So, by (5.3),

$$\begin{aligned}|x_i^{k(\nu+1)} - x_i^*| &\le \Gamma L(\|\bar{x}_{i-1}^{k+1} - \bar{x}_{i-1}^*\| + |x_i^k - x_i^*| \\ &\qquad + |x_i^{k(\nu)} - x_i^*|)|x_i^{k(\nu)} - x_i^*| + \Gamma\alpha\sum_{j=1}^{i-1}|x_j^{k+1} - x_j^*| \\ &\le \Gamma L\Bigg(\sum_{j=1}^{i-1}|x_j^{k+1} - x_j^*| + |x_i^k - x_i^*| + c_{i,\nu}\|\bar{x}_i^k - \bar{x}_i^*\|^{\nu+1}\Bigg) \\ &\qquad \times c_{i,\nu}\|\bar{x}_i^k - \bar{x}_i^*\|^{\nu+1} + \Gamma\alpha\sum_{j=1}^{i-1}|x_j^{k(q)} - x_j^*| \\ &\le \Gamma L\Bigg(\sum_{j=1}^{i-1}c_{j,q}\|\bar{x}_j^k - \bar{x}_j^*\|^{q+1} + |x_i^k - x_i^*| \\ &\qquad + c_{i,\nu}\|\bar{x}_i^k - \bar{x}_i^*\|^{\nu+1}\Bigg)c_{i,\nu}\|\bar{x}_i^k - \bar{x}_i^*\|^{\nu+1} \\ &\quad + \Gamma\alpha\sum_{j=1}^{i-1}c_{j,q}\|\bar{x}_j^k - \bar{x}_j^*\|^{q+1} \\ &\le \Gamma L\Bigg(\sum_{j=1}^{i-1}c_{j,q}\|\bar{x}_i^k - \bar{x}_i^*\|^{q} + 1 + c_{i,\nu}\|\bar{x}_i^k - \bar{x}_i^*\|^{\nu}\Bigg)c_{i,\nu}\|\bar{x}_i^k - \bar{x}_i^*\|^{\nu+2} \\ &\quad + \Gamma\alpha\Bigg(\sum_{j=1}^{i-1}c_{j,q}\Bigg)(\|\bar{x}_i^k - \bar{x}_i^*\|^{q-(\nu+1)})\|\bar{x}_i^k - \bar{x}_i^*\|^{\nu+2}.\end{aligned}$$

Therefore, the inductive step is complete, defining

$$c_{i,\nu+1} = \Gamma L\Bigg(\Bigg(\sum_{j=1}^{i-1}c_{j,q}\Bigg)\epsilon_3^q + 1 + c_{i,\nu}\epsilon_3^{\nu}\Bigg)c_{i,\nu}$$

(A.15)
$$+\Gamma\alpha\Big(\sum_{j=1}^{i-1} c_{j,q}\Big)\epsilon_3^{q-(\nu+1)}.$$

Let us now prove (A.14) for the case where B_i^k is chosen by (4.8) (the modified Gauss–Seidel–Newton). We also proceed by induction on i. As before, let us prove (A.14) first for $i = 1$. For $i = 1$ and $\nu = 0$, (A.14) is also trivial. Let us prove the inductive step on ν.

$$\begin{aligned}
|x_1^{k(\nu+1)} - x_1^*| &\le \Gamma(|J_1(x_1^k) - J_1(x_1^*)| \\
&\qquad + L|x_1^{k(\nu)} - x_1^*|)|x_1^{k(\nu)} - x_1^*| \\
&\le \Gamma L(\|x_1^k - x_1^*\| + L|x_1^{k(\nu)} - x_1^*|)|x_1^{k(\nu)} - x_1^*| \\
&\le \Gamma L(\|x_1^k - x_1^*\| + Lc_{1,\nu}\|x_1^k - x_1^*\|^{\nu+1})c_{1,\nu}\|x_1^k - x_1^*\|^{\nu+1} \\
&\le \Gamma L(1 + Lc_{1,\nu}\epsilon_3^\nu)c_{1,\nu}\|x_1^k - x_1^*\|^{\nu+2}.
\end{aligned}$$

So, in this case, we must define

(A.16)
$$c_{1,\nu+1} = \Gamma L(1 + Lc_{1,\nu}\epsilon_3^\nu)c_{i,\nu}.$$

Now let us prove (A.14) for the choice (3.8) for a generic $i \in \{1,\dots,m\}$, assuming that $|x_j^{k(\nu)} - x_j^*| \le c_{j,\nu}\|\bar{x}_j^k - \bar{x}_j^*\|^{\nu+1}$ holds for all $j \in \{1,\dots,i-1\}$. For $\nu = 0$, (A.14) is again trivial. For $\nu + 1$, we have that

$$\begin{aligned}
|x_i^{k(\nu+1)} - x_i^*| &\le \Gamma(|J_i(\bar{x}_i^k) - J_i(\bar{x}_i^*)| \\
&\qquad + L|x_i^{k(\nu)} - x_i^*|)|x_i^{k(\nu)} - x_i^*| + \Gamma\alpha\|\bar{x}_{i-1}^{k+1} - \bar{x}_{i-1}^*\| \\
&\le \Gamma L(\|\bar{x}_i^k - \bar{x}_i^*\| + L|x_i^{k(\nu)} - x_i^*|)|x_i^{k(\nu)} - x_i^*| + \Gamma\alpha\|\bar{x}_{i-1}^{k+1} - \bar{x}_{i-1}^*\| \\
&\le \Gamma L(\|\bar{x}_i^k - \bar{x}_i^*\| + Lc_{i,\nu}\|\bar{x}_i^k - \bar{x}_i^*\|^{\nu+1})c_{i,\nu}\|\bar{x}_i^k - \bar{x}_i^*\|^{\nu+1} \\
&\qquad + \Gamma\alpha\Big(\sum_{j=1}^{i-1} c_{j,q}\Big)\epsilon_3^{q-(\nu+1)}\|\bar{x}_i^k - \bar{x}_i^*\|^{\nu+2} \\
&\le \Big(\Gamma L(1 + Lc_{i,\nu}\epsilon_3^\nu)c_{i,\nu} + \Gamma\alpha\Big(\sum_{j=1}^{i-1} c_{j,q}\Big)\epsilon_3^{q-(\nu+1)}\Big)\|\bar{x}_i^k - \bar{x}_i^*\|^{\nu+2}.
\end{aligned}$$

So, in this case, we must define

(A.17)
$$c_{i,\nu+1} = \Gamma L(1 + Lc_{i,\nu}\epsilon_3^\nu)c_{i,\nu} + \Gamma\alpha\Big(\sum_{j=1}^{i-1} c_{j,q}\Big)\epsilon_3^{q-(\nu+1)}.$$

Therefore, the inductive proof is complete and (A.14) holds both for the choice (3.7) and for the choice (3.8) of B_i^k. Now, defining $c_1' = \max_{i,\nu}\{c_{i,\nu}\}$, we have that

(A.18)
$$|x_i^{k(\nu)} - x_i^*| \le c_1'\|\bar{x}_i^k - \bar{x}_i^*\|^{\nu+1}$$

for $\nu = 1,\dots,q$, $i = 1,\dots,m$, $k = 0,1,2,\dots$. So,

(A.19)
$$|x_i^{k+1} - x_i^*| = |x_i^{k(q)} - x_i^*| \le c_1'\|\bar{x}_i^k - \bar{x}_i^*\|^{q+1} \le c_1'\|x^k - x^*\|^{q+1}$$

for $i = 1, \ldots, m$. Adding the inequalities (A.19) for $i = 1, \ldots, m$, we obtain

$$\|x^{k+1} - x^*\| \leq mc_1' \|x^k - x^*\|^{q+1}.$$

Hence, (5.12), (5.13), and (5.14) follow, defining $c_1 = mc_1'$. This completes the proof of the theorem. □

We need an auxiliary lemma to prove Theorem 5.4.

LEMMA A2. *Let $e_i^k, i = 1, \ldots, m, k = 0, 1, 2, \ldots$, $\beta_1, \beta_2, \beta_3 > 0$ be real numbers such that $e_i^k \geq 0$ for all $i = 1, \ldots, m, k = 0, 1, 2, \ldots$, and*

$$e_i^{k+1} \leq \beta_1 (e_i^k)^2 + (\beta_2 + \beta_3 e_i^k) \sum_{j=1}^{i-1} e_j^k. \tag{A.20}$$

Let $\epsilon > 0$. There exists $\epsilon' \in (0, \epsilon]$ such that if

$$\sum_{i=1}^{m} e_i^0 \leq \epsilon' \tag{A.21}$$

we have

$$\sum_{i=1}^{m} e_i^k \leq \epsilon \tag{A.22}$$

for all $k = 0, 1, 2, \ldots$ and

$$\lim_{k \to \infty} e_i^k = 0 \tag{A.23}$$

for $i = 1, \ldots, m$. Moreover, there exists $c > 0$ such that

$$e_i^{k+1} \leq c[(e_i^k)^2 + \cdots + (e_1^{k-i+1})^2] \tag{A.24}$$

for all i, k such that $k \geq m - 1, i = 1, \ldots, m$.

Proof. Define

$$w_i^k = e_i^{k+i-1} \tag{A.25}$$

for $i = 1, \ldots, m$, $k = 0, 1, 2, \ldots$,

$$w^k = (w_1^k, \ldots, w_m^k). \tag{A.26}$$

Let us consider a fixed k. We will prove that

$$w_i^{k+\nu} \leq \sum_{\lambda \in \Upsilon(i,\nu)} \phi_\lambda(w_1^k, \ldots, w_i^k), \qquad \nu = 1, \ldots, m, \tag{A.27}$$

where $\phi_\lambda(y_1, \ldots, y_i)$ is a monomial of degree at least 2 whose coefficients are positive and independent of k, $\Upsilon(i, \nu)$ is a set of indices, and $\vartheta(i, \nu)$ is the number of monomials that occur in (A.27).

We proceed by induction on i. So, let us first prove (A.27) for $i = 1$. In fact, by (A.20), we have that

$$w_1^{k+\nu} = e_1^{k+\nu} \leq \beta_1^\nu (e_1^k)^{2\nu} = \beta_1^\nu (w_1^k)^{2\nu}. \tag{A.28}$$

So, (A.27) holds for $i = 1$ with $\vartheta(1,\nu) = 1$ for all ν, and $\phi_\lambda(w_1^k) = \beta_1^\nu (w_1^k)^{2\nu}$.

Now, let us assume that $w_j^{k+\nu} \le \sum_{\lambda\in\Upsilon(j,\nu)} \phi_\lambda(w_1^k,\dots,w_j^k)$ for all $j = 1,\dots,i-1$, $\nu \ge 1$. Using this, we are going to prove (A.27). We proceed by induction on ν. If $\nu = 1$, we have, by (A.20) and the inductive hypothesis, that

$$\begin{aligned} w_i^{k+1} = e_i^{k+i} &\le \beta_1 (e_i^{k+i-1})^2 + (\beta_2 + \beta_3 e_i^{k+i-1}) \sum_{j=1}^{i-1} e_j^{k+i-1} \\ &\le \beta_1 (e_i^{k+i-1})^2 + (\beta_2 + \beta_3 e_i^{k+i-1}) \left(\sum_{j=1}^{i-1} e_j^{k+i-1} \right) \\ &= \beta_1 (w_i^k)^2 + (\beta_2 + \beta_3 w_i^k) \left(\sum_{j=1}^{i-1} w_j^{k+j} \right) \\ &= \beta_1 (w_i^k)^2 + (\beta_2 + \beta_3 w_i^k) \left(\sum_{j=1}^{i-1} \sum_{\lambda\in\Upsilon(j,j)} \phi_\lambda(w_1^k,\dots,w_j^k) \right). \end{aligned}$$

The right-hand side of the last inequality has the form of the right-hand side of (A.27). So, (A.27) is proved for $\nu = 1$. Finally, assume that

$$w_j^{k+\ell} \le \sum_{\lambda\in\Upsilon(j,\ell)} \phi_\lambda(w_1^k,\dots,w_j^k)$$

for all $j = 1,\dots,i-1$ and for $j = i$, $\ell = 1,\dots,\nu-1$. Then, by (A.20),

$$\begin{aligned} w_i^{k+\nu} = e_i^{k+\nu+i-1} &\le \beta_1 (e_i^{k+\nu+i-2})^2 + (\beta_2 + \beta_3 e_i^{k+\nu+i-2}) \sum_{j=1}^{i-1} e_j^{k+\nu+i-2} \\ &\le \beta_1 (w_i^{k+\nu-1})^2 + (\beta_2 + \beta_3 w_i^{k+\nu-1}) \sum_{j=1}^{i-1} w_j^{k+\nu+j-1} \\ &\le \beta_1 \left[\sum_{\lambda\in\Upsilon(i,\nu-1)} \phi_\lambda(w_1^k,\dots,w_i^k) \right]^2 \\ &\quad + \left(\beta_2 + \beta_3 \sum_{\lambda\in\Upsilon(i,\nu-1)} \phi_\lambda(w_1^k,\dots,w_i^k) \right) \left(\sum_{j=1}^{i-1} \sum_{\lambda\in\Upsilon(j,\nu+j-1)} \phi_\lambda(w_1^k,\dots,w_j^k) \right) \\ &= \sum_{\lambda\in\Upsilon(i,\nu)} \phi_\lambda(w_1^k,\dots,w_i^k) \end{aligned}$$

for a suitable definition of $\Upsilon(i,\nu)$. From the construction, we easily see that the coefficients and indices involved in the monomials at (A.26) are independent of k.

In particular, for $\nu = 1$, we have that

$$w_i^{k+1} \le \sum_{\lambda\in\Upsilon(i,1)} \phi_\lambda(w_1^k,\dots,w_i^k) \tag{A.29}$$

for all $i = 1,\dots,m$, $k = 0,1,2,\dots$. But, since $w_j^k \le (w_1^2 + \cdots + w_i^2)^{\frac{1}{2}} \le \|w^k\|_2$ for all $i = 1,\dots,m$, $j = 1,\dots,i$, (A.28) implies that

$$w_i^{k+1} \le (w_1^2 + \cdots + w_i^2) P(\|w^k\|_2), \tag{A.30}$$

where P is a polynomial. By (A.30) and (A.20), $\|w^0\|$ can be made as small as desired if $\|e^0\|$ is small enough. Therefore, it holds that (A.23) and (A.24) follow from (A.30) and (A.20). □

Proof of Theorem 5.4. Define

$$e_i^k = |x_i^k - x_i^*| \tag{A.31}$$

for $i = 1, \ldots, m$, $k = 0, 1, 2, \ldots$. Then by (5.5) and (5.3), we have, if $\|x^k - x^*\| \leq \epsilon_1$, that

(A.32)

$$\begin{aligned}
e_i^{k+1} &= |x_i^{k+1} - x_i^*| = |x_i^k - x_i^* - J_i(x_1^k, \ldots, x_i^k)^{-1} F_i(x_1^k, \ldots, x_i^k)| \\
&\leq |x_i^k - x_i^* - J_i(\bar{x}_i^k)^{-1} F_i(\bar{x}_{i-1}^*, x_i^k)| + |J_i(\bar{x}_i^k)^{-1}| \cdot |F_i(\bar{x}_{i-1}^*, x_i^k) - F_i(\bar{x}_i^k)| \\
&\leq |x_i^k - x_i^* - J_i(\bar{x}_i^k)^{-1} \left(\int_0^1 J_i(\bar{x}_{i-1}^*, x_i^* + t(x_i^k - x_i^*)) dt \right) (x_i^k - x_i^*)| \\
&\quad + \alpha |J_i(\bar{x}_i^k)^{-1}| \cdot \|\bar{x}_{i-1}^k - \bar{x}_{i-1}^*\| \\
&\leq \Gamma |J_i(\bar{x}_i^k) - J_i(\bar{x}_i^*)| |x_i^k - x_i^*| + \Gamma L |x_i^k - x_i^*|^2 + \alpha\Gamma \|\bar{x}_{i-1}^k - \bar{x}_{i-1}^*\| \\
&\leq \Gamma L \|\bar{x}_i^k - \bar{x}_i^*\| \cdot |x_i^k - x_i^*| + \Gamma L |x_i^k - x_i^*|^2 + \alpha\Gamma \|\bar{x}_{i-1}^k - \bar{x}_{i-1}^*\| \\
&\leq 2\Gamma L |x_i^k - x_i^*|^2 + (\alpha\Gamma + \Gamma L |x_i^k - x_i^*|) \sum_{j=1}^{i-1} |x_j^k - x_j^*| \\
&= 2\Gamma L (e_i^k)^2 + (\alpha\Gamma + \Gamma L e_i^k) \sum_{j=1}^{i-1} e_j^k.
\end{aligned}$$

Hence, by (A.31) and (A.32), the inequality (A.20) holds if $\|x^k - x^*\|$ is small enough. Therefore, the thesis follows by Lemma A2 using a straightforward inductive argument. □

REFERENCES

[1] C. G. BROYDEN, *A class of methods for solving nonlinear simultaneous equations*, Math. Comp., 19 (1965), pp. 577–593.

[2] T. F. COLEMAN, B. S. GARBOW, AND J. J. MORÉ, *Software for estimating sparse Jacobian matrices*, ACM Trans. Math. Software, 11 (1984), pp. 363–378.

[3] T. F. COLEMAN AND J. J. MORÉ, *Estimation of sparse Jacobian matrices and graph coloring problems*, SIAM J. Numer. Anal., 20 (1983), pp. 187–209.

[4] A. R. CURTIS, M. J. D. POWELL, AND J. K. REID, *On the estimation of sparse Jacobian matrices*, J. Inst. Math. Appl., 13 (1974), pp. 117–120.

[5] J. E. DENNIS, JR. AND R. B. SCHNABEL, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, Englewood Cliffs, NJ, 1983.

[6] I. S. DUFF, *On permutations to block triangular form*, J. Inst. Math. Appl., 19 (1977), pp. 339–342.

[7] ———, MA28—*a set of Fortran subroutines for sparse unsymmetric linear equations*, Report, AERE R8730, Her Majesty's Stationery Office, London, 1977.

[8] I. S. DUFF, A. M. ERISMAN, AND J. K. REID, *Direct Methods for Sparse Matrices*, Oxford Scientific Publications, 1989.

[9] J. ERICKSON, *A note on the decomposition of systems of sparse non-linear equations*, BIT, 16 (1976), pp. 462–465.

[10] F. HARARY, *Sparse matrices and graph theory*, in Large Sparse Sets of Linear Equations, J. K. Reid, ed., Academic Press, New York, 1971.

[11] T. LOVETT AND S. THAKKAR, *The symmetry multiprocessor system*, in Proceedings of 1988 International Conference of Parallel Processing, August 1988, CRC Press, Boca Raton, FL, pp. 303–310.

[12] J. MORÉ AND M. COSNARD, *Numerical solution of nonlinear equations*, ACM Trans. Math. Software, 5 (1979), pp. 64–85.
[13] J. MORÉ, B. S. GARBOW, AND K. E. HILLSTROM, *Testing unconstrained optimization software*, ACM Trans. Math. Software, 7 (1981), pp. 17–41.
[14] J. M. ORTEGA AND W. G. RHEINBOLDT, *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, New York, 1970.
[15] D. V. STEWARD, *On a approach to techniques for the analysis of the structure of large systems of equations*, SIAM Rev., 4 (1962), pp. 321–342.
[16] R. E. TARJAN, *Depth-first search and linear graph algorithms*, SIAM J. Comput., 1 (1972), pp. 146–160.

SIAM J. OPTIMIZATION
Vol. 4, No. 2, pp. 383–392, May 1994
© 1994 Society for Industrial and Applied Mathematics
009

EXTENSION OF HOFFMAN'S ERROR BOUND TO POLYNOMIAL SYSTEMS*

XIAO-DONG LUO† AND ZHI-QUAN LUO‡

Abstract. Given any linear system (defined by linear inequalities/equalities) in $\Re^n$, Hoffman's error bound says that the distance from any point $x \in \Re^n$ to the solution set of the linear system is bounded by a constant (independent of x) times a certain residual function evaluated at x. This paper considers arbitrary (possibly nonconvex) polynomial systems (defined by polynomial inequalities/equalities). It is shown that the solution sets of such systems are, in general, Hölder continuous as the right-hand side changes, in contrast to the Lipschitzian continuity for the linear case. Also, for any convex quadratic inequality system possessing an interior solution, the authors show that its solution set behaves in a Lipschitzian manner as the right-hand side varies. This result sharpens the earlier error bound results by Mangasarian and Robinson for general convex differentiable inequality systems, which, in addition to the interiority assumption, require either an asymptotic constraint qualification condition or that the solution set be bounded.

Key words. Hoffman's error bound, polynomial inequality systems, Lipschitzian continuity

AMS subject classifications. 90C31, 65F35, 15A39

1. Introduction.

Consider the following inequality system:

$$f_1(x) \le 0,\ f_2(x) \le 0, \ldots,\ f_m(x) \le 0,\ g_1(x) = 0,\ g_2(x) = 0, \ldots,\ g_k(x) = 0, \tag{1.1}$$

where $x \in \Re^n$ and each f_i, g_j is a differentiable function of x. We let P denote the solution set of (1.1) and assume throughout that P is nonempty. We let $f : \Re^n \mapsto \Re^m$ be the vector function whose ith component function is f_i and let g be defined likewise. In the special case where f and g are affine linear mappings, i.e., $f(x) = Ax + a$ and $g(x) = Bx + b$, for some matrices A, B and some vectors a, b of appropriate dimensions, Hoffman [Hof52] showed the following fundamental result: there exists some $\tau > 0$ depending on A and B only such that

$$\operatorname{dist}(x, P) \le \tau(\|[Ax + a]_+\| + \|Bx + b\|) \quad \forall x \in \Re^n, \tag{1.2}$$

where $\operatorname{dist}(\cdot, \cdot)$ denotes the Euclidean distance function between two sets, $\|\cdot\|$ denotes the usual Euclidean norm in $\Re^n$, and $[\cdot]_+$ denotes the positive part of a vector. In other words, the distance from any vector $x \in \Re^n$ to P can be bounded from above by the amount of violation of the constraints at x. The relation (1.2) basically amounts to the Lipschitzian continuity of the solution set of a linear system as the right-hand side is perturbed. Error bound (1.2) and its relatives have been studied quite extensively, and they have often been used in the sensitivity analysis of linearly constrained optimization problems such as linear programs, linear complementarity problems (see [MaD88] and [Pan87]). Recently, these error bounds have also found many important applications in the analysis of descent algorithms for degenerate convex minimization problems subject to linear constraints (see [LuT92a] and [LuT92b]).

* Received by the editors June 22, 1992; accepted for publication (in revised form) March 26, 1993.

† Operations Research Center, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139 (xluo@athena.mit.edu).

‡ Communications Research Laboratory, McMaster University, Hamilton, Ontario L8S 4K1, Canada (luozq@mcmail.cis.mcmaster.ca). This author's research was supported by Natural Sciences and Engineering Research Council of Canada grant OPG0090391.

The objective of this paper is to extend Hoffman's result to the general case where f is an arbitrary polynomial mapping (i.e., each f_i is a polynomial but not necessarily convex). To motivate our result, we consider the following simple example.

Example 1.1. Let $f : \Re \mapsto \Re$ be given by $f(x) = x^2$. Clearly the solution set $P = \{x \mid f(x) \leq 0\}$ consists of a single point: the origin. Given any point $x \in \Re$ with $x \neq 0$, we have $\text{dist}(x, P) = |x|$. However, $\|[f(x)]_+\|$, the amount of violation of the constraint $x^2 \leq 0$, is equal to x^2. Therefore, the bound (1.2) cannot hold for x near the origin. Instead, we have in this example that $\text{dist}(x, P) \leq \|[f(x)]_+\|^{1/2}$.

Example 1.1 implies that Hoffman's result (1.2) does not hold for general polynomial mappings. That is, the Lipschitzian continuity of the solution set is lost when we depart from the linear case. It suggests, instead, the following generalization of (1.2):

$$\text{dist}(x, P) \leq \tau \left(\|[f(x)]_+\| + \|g(x)\|\right)^{\gamma}, \tag{1.3}$$

for all $x \in \Re^n$, where τ and γ are some positive constants depending on the coefficients and the degrees of the polynomials $f_1, \ldots, f_m$, $g_1, \ldots, g_k$ only; in other words, the solution set of a nonlinear polynomial system is Hölder continuous with respect to the right-hand side. Indeed, Hörmander [Hor58] (see also [Gor61] and the appendix of [Hor69] for a related analysis) was the first to establish (1.3) for the case $f(x)$ is a single polynomial, though an extra multiplicative factor $(1 + \|x\|)^{\gamma'}$ is needed in the right-hand side ($\gamma' \geq 0$ is a constant). (Hörmander used this result to prove the existence of a certain fundamental solution for a class of partial differential equations with constant coefficients.) Other related results include that of Robinson [Rob75], who showed an error bound of the form (1.3) with $\gamma = 1$ for general convex differentiable inequality systems (i.e., each f_i is convex differentiable), under the assumption that P is *bounded* and has an nonempty interior. Later, Mangasarian [Man85] replaced the boundedness assumption by an asymptotic constraint qualification condition. Unfortunately, Mangasarian's condition is quite complex and difficult to verify in practice.

In this paper, we first generalize Hörmander's result to the case of multiple polynomial equalities/inequalities (see §2). Then, we show that for the convex quadratic polynomial inequality systems (i.e., each f_i is a convex quadratic polynomial and each g_j is linear), the above error bound can be sharpened substantially: (1.3) holds with $\gamma = 1$ and without the extra multiplicative factor (see §3), provided that P satisfies a nonempty interior assumption (see Assumption 3.1). Our results indicate that, for the convex quadratic inequality systems, neither the compactness condition of Robinson [Rob75] nor the asymptotic constraint qualification condition of Mangasarian [Man85] is needed.

We briefly describe our notation below. All vectors are column vectors and superscript T denotes transpose. We denote by $\langle \cdot, \cdot \rangle$ the Euclidean inner product and by $\|\cdot\|$ the usual Euclidean norm. For any $k \times l$ matrix B, we denote by $\|B\|$ the matrix norm of B as induced by the vector norm $\|\cdot\|$ (i.e., $\|B\| = \max_{\|x\|=1} \|Bx\|$), by B_i the ith row of B, and, for any $I \subseteq \{1, \ldots, k\}$, by B_I the submatrix of B obtained by removing all rows $i \notin I$ of B. Analogously, for any $x \in \Re^k$, we denote by x_i the ith coordinate of x, and, for any $I \subseteq \{1, \ldots, k\}$, by x_I the vector with components x_i, $i \in I$ (with the x_i's arranged in the same order as in x).

2. An error bound to polynomial systems. In this section, we show that the error bound (1.3) indeed holds, except that an extra factor of $(1 + \|x\|)^{\gamma'}$ ($\gamma' \geq 0$ is a constant) has to be introduced (see Theorem 2.2). Our result is based on the

following error bound result of Hörmander, which considered the case where f is a single polynomial.

THEOREM 2.1. *Let $f(x)$ be a real polynomial of the variables $x_1, \ldots, x_n$. Let P denote the set of real zeros of f, that is,*

$$P = \{x \mid f(x) = 0,\ x = (x_1, \ldots, x_n)\},$$

and suppose P is nonempty. Then there exist positive constants τ, γ and a (possibly negative) constant γ' such that

$$\operatorname{dist}(x, P) \leq \tau(1 + \|x\|)^{\gamma'} |f(x)|^{\gamma} \quad \forall x \in \Re^n. \tag{2.1}$$

In what follows, we shall extend Theorem 2.1 to the system of polynomial equations and inequalities. The idea is to add slack variables to the system (1.1), convert it to an equivalent system with a single polynomial equation, and then apply Theorem 2.1 to this new system.

THEOREM 2.2. *Let P denote the set of x in $\Re^n$ satisfying $f_1(x) \leq 0, \ldots, f_m(x) \leq 0$, $g_1(x) = 0, \ldots, g_k(x) = 0$, where each f_i and g_j is a polynomial with real coefficients. Suppose that P is nonempty. Then, there exist constants $\tau > 0$, $\gamma > 0$, and $\gamma' \geq 0$ such that*

$$\operatorname{dist}(x, P) \leq \tau(1 + \|x\|)^{\gamma'} (\|[f(x)]_+\| + \|g(x)\|)^{\gamma} \quad \forall x \in \Re^n. \tag{2.2}$$

Here $f(x) = (f_1(x), \ldots, f_m(x))^T$ and $g(x) = (g_1(x), \ldots, g_k(x))^T$.

Proof. Consider the polynomial $h : \Re^{n+m} \mapsto \Re$ given by

$$h(x, z) = (f_1(x) + z_1^2)^2 + \cdots + (f_m(x) + z_m^2)^2 + g_1^2(x) + \cdots + g_k^2(x), \tag{2.3}$$

where $z = (z_1, \ldots, z_m)^T$. Let $\bar{P} \subset \Re^{n+m}$ denote the set of (x, z) such that $h(x, z) = 0$. It is easily seen that

$$x \in P \Longleftrightarrow (x, z) \in \bar{P} \text{ with } z_i = \sqrt{[-f_i(x)]_+},\ i = 1, \ldots, m. \tag{2.4}$$

Since P is nonempty, it follows that $\bar{P}$ is also nonempty.

By Theorem 2.1, there exist some constants $\theta > 0$, $\mu > 0$, and $\mu' \geq 0$ such that $\operatorname{dist}((x, z), \bar{P}) \leq \theta(1 + \|(x, z)\|)^{\mu'} |h(x, z)|^{\mu}$ for all $(x, z) \in \Re^{n+m}$. (We assume here that μ' is positive.) Given any $x \in \Re^n$, let $z \in \Re^m$ be given by $z_i = \sqrt{[-f_i(x)]_+}$, $i = 1, \ldots, m$. Let $(x^*, z^*) \in \bar{P}$ be such that $\|(x, z) - (x^*, z^*)\| = \operatorname{dist}((x, z), \bar{P})$. Then we have

$$\begin{aligned}
\|x - x^*\| &\leq \|(x, z) - (x^*, z^*)\| \\
&= \operatorname{dist}((x, z), \bar{P}) \\
&\leq \theta(1 + \|(x, z)\|)^{\mu'} |h(x, z)|^{\mu} \\
&\leq \theta(1 + \|x\| + \|z\|)^{\mu'} \left([f_1(x)]_+^2 + \cdots + [f_m(x)]_+^2 + g_1^2(x) + \cdots + g_k^2(x)\right)^{\mu} \qquad (2.5) \\
&\leq \tau(1 + \|x\|)^{\gamma'} \left(\|[f(x)]_+\|^2 + \|g(x)\|^2\right)^{\mu} \qquad (2.6) \\
&\leq \tau(1 + \|x\|)^{\gamma'} (\|[f(x)]_+\| + \|g(x)\|)^{\gamma},
\end{aligned}$$

where (2.5) is due to (2.3) and the fact $f_i(x) + z_i^2 = f_i(x) + [-f_i(x)]_+ = [f_i(x)]_+$, $i = 1, \ldots, m$, and (2.6) follows from $z_i^2 \leq |f_i(x)| \leq c(1 + \|x\|)^L$, $i = 1, \ldots, m$, for some

positive constants c and L. Next set τ and γ' sufficiently large; the last step follows from setting $\gamma = 2\mu$. This completes the proof of the desired result. □

The error bound given in Theorem 2.2 is weaker than Hoffman's error bound (1.2) in several respects. First, it has an extra factor of $(1 + \|x\|)^{\gamma'}$. We can remove this extra factor by restricting the error bound to a bounded region.

COROLLARY 2.3. *Let $f_1, \ldots, f_m$, $g_1, \ldots, g_k$, and P be given as in Theorem 2.2. Let ρ be a positive number such that P contains an element x with $\|x\| \leq \rho$. Then there holds*

$$\operatorname{dist}(x, P) \leq \tau' \left(\|[f(x)]_+\| + \|g(x)\|\right)^{\gamma} \quad \forall x \text{ with } \|x\| \leq \rho,$$

for some $\tau' > 0$ and $\gamma > 0$.

Proof. Apply Theorem 2.2, and let $\tau' = \tau(1 + \rho)^{\gamma'}$ (cf. $\gamma' \geq 0$). □

The following example indicates that in general the multiplicative factor $(1 + \|x\|)^{\gamma'}$ cannot be removed from the error bound (2.2).

Example 2.1. Consider the following quadratic system in $\Re^2$:

$$x_1 x_2 = 0, \quad x_1 \geq 0, \quad x_2 \geq 1.$$

Clearly, the solution set to the above system is $P = \{(0, x_2) \mid x_2 \geq 1\}$. Consider the points $(t, 0)$ on the x_1-axis with $t \geq 0$. Then the residual evaluated at $(t, 0)$ is equal to 1. However, the distance from $(t, 0)$ to P is at least t, which is unbounded as $t \to \infty$. Thus, the error bound (2.2) cannot hold for all $x \in \Re^n$ if the multiplicative factor $(1 + \|x\|)^{\gamma'}$ is removed.

The error bound (2.2) is weaker than Hoffman's error bound (1.2) in another respect: the exponent γ in (2.2) is typically a small (and unknown) positive number, in contrast to the exponent $\gamma = 1$ in the error bound (1.2). This seems inevitable, however, since Example 1.1 has indicated that γ is strictly smaller than 1 in general. Recently Luo and Pang [LuP93] proved that (2.2) holds with $\gamma = \frac{1}{2}$ for a class of quadratic polynomial systems satisfying a certain nonnegativity assumption. In the next section, we will prove that $\gamma = 1$ is a valid exponent for convex quadratic polynomial inequality systems satisfying a Slater condition.

3. Error bound for convex quadratic inequality systems. In this section, we consider a convex quadratic inequality system given by

$$f_1(x) \leq 0, \; f_2(x) \leq 0, \ldots, \; f_m(x) \leq 0, \tag{3.1}$$

where $f_i(x) = \langle x, Q^i x\rangle/2 + \langle b^i, x\rangle - c_i$, $i = 1, 2, \ldots, m$, with each Q^i an $n \times n$ positive semidefinite matrix, each b^i an n-vector, and each c_i a scalar. We shall use N (respectively, L) to denote the set of indices i for which f_i is nonlinear (respectively, linear). Let P denote the solution set of (3.1). We make the following assumption regarding the quadratic inequality system (3.1).

Assumption 3.1. There exists some $x^* \in P$ such that $f_i(x^*) < 0$ for all $i \in N$. In other words, x^* satisfies the *nonlinear* constraints of (3.1) with strict inequality.

Under Assumption 3.1 we shall prove that an analogue of Hoffman's error bound holds for the convex quadratic inequality system (3.1). In particular, we have the following.

THEOREM 3.1. *Suppose that Assumption* 3.1 *holds. Then there exists some positive scalar τ depending on the problem data only such that*

$$\operatorname{dist}(x, P) \leq \tau \|[f(x)]_+\| \quad \forall x \in \Re^n, \tag{3.2}$$

where $f(x) = (f_1(x), f_2(x), \ldots, f_m(x))^T$.

As noted before, an error bound identical to (3.2) has been established by Mangasarian [Man85] for general convex differentiable inequality systems, under an assumption similar to Assumption 3.1 plus an additional asymptotic constraint qualification assumption.

THEOREM 3.2 (Mangasarian). *Let* f *be a differentiable convex function from* $\Re^n$ *to* $\Re^m$. *Let* $N \cup L$ *be a partition of* $\{1, \ldots, m\}$ *such that* f_N *is nonlinear and* f_L *is linear, and let* $P = \{x \mid f(x) \leq 0\}$. *Suppose there exists some* $x^* \in P$ *such that* $f_N(x^*) < 0$ *and the constant* μ *defined by*

(3.3)

$$\mu = \sqrt{n}\sup_{\lambda,x,I}\{\|\lambda_I\| \mid x \in P,\ \lambda_I > 0,\ f_I(x) = 0,$$
$$\|\textstyle\sum_{i\in I} \lambda_i \nabla f_i(x)\| = 1,\ \nabla f_i(x) \text{ linearly independent, } I \subset \{1, \ldots, m\}\}$$

is finite. Then

$$\operatorname{dist}(x, S) \leq \mu\|[f(x)]_+\| \quad \forall x \in \Re^n. \tag{3.4}$$

Moreover, when a certain asymptotic constraint qualification condition is satisfied, the constant μ *defined by* (3.3) *is finite.*

Unfortunately, neither the condition (3.3) nor the asymptotic constraint qualification condition (see [Man85] for details) is easy to verify in practice. What we will show next is that, for the convex quadratic inequality system (3.1), the constant μ is always finite and that Mangasarian's asymptotic constraint qualification condition can be removed. The proof of this, which amounts to a proof of Theorem 3.1, consists of three lemmas—the first two of which are simple properties of symmetric positive semidefinite matrices.

LEMMA 3.3. *For any* $n \times n$ *symmetric positive semidefinite matrix* Q, *there exist two positive constants* u *and* v *such that*

$$u\|Qx\|^2 \leq \langle x, Qx\rangle \leq v\|Qx\|^2,$$

for all $x \in \Re^n$.

The proof of Lemma 3.3 is straightforward and therefore is omitted.

LEMMA 3.4. *Suppose that* $Q^1, \ldots, Q^k$ *are some* $n \times n$ *symmetric positive semidefinite matrices. Let* θ *be a positive constant. Then, for all* $\lambda_1, \ldots, \lambda_k$ *with* $\lambda_i \geq \theta$, $i = 1, \ldots, k$, *there holds*

$$\left\|\lambda_1 Q^1 x + \cdots + \lambda_k Q^k x\right\|^2 \geq \tau\left(\|Q^1 x\|^2 + \cdots + \|Q^k x\|^2\right) \quad \forall x \in \Re^n,$$

where τ *is a positive constant depending on* θ *and* $Q^1, \ldots, Q^k$ *only.*

Proof. Fix any $x \in \Re^n$. We take an $y \in \Re^n$ with the property

$$\|y\| \leq M(\|Q^1 y\| + \cdots + \|Q^k y\|), \qquad Q^1 y = Q^1 x, \ldots, Q^k y = Q^k x, \tag{3.5}$$

where M is some constant depending on $Q^1, \ldots, Q^k$ only. (The existence of such y and M is guaranteed by, for example, Hoffman's error bound (1.2).) By Lemma 3.3, there exists some $u > 0$ depending on $Q^1, \ldots, Q^k$ only such that $\|Q^i y\|^2 \leq \langle y, Q^i y\rangle/u$, for all i. Thus, we have

$$\begin{aligned}
\|Q^i y\|^2 &\leq \langle y, Q^i y\rangle/u \\
&\leq \frac{1}{\theta u}\langle y, (\lambda_1 Q^1 + \cdots + \lambda_k Q^k) y\rangle \\
&\leq \frac{M}{\theta u}(\|Q^1 y\| + \cdots + \|Q^k y\|)\left\|(\lambda_1 Q^1 + \cdots + \lambda_k Q^k) y\right\| \quad \forall i,
\end{aligned}$$

where the second step follows from $\lambda_i \geq \theta$ and $Q^i \geq 0$ (i.e., positive semidefinite), $i = 1, \ldots, k$, and the last step is due to the Cauchy–Schwarz inequality and (3.5). Summing up the above relation for all i and using the Cauchy–Schwarz inequality again yields

$$\sum_{i=1}^{k} \|Q^i y\|^2 \leq \frac{Mk}{\theta u}(\|Q^1 y\| + \cdots + \|Q^k y\|) \left\|(\lambda_1 Q^1 + \cdots + \lambda_k Q^k) y\right\|$$

$$\leq \frac{Mk^{3/2}}{\theta u}(\|Q^1 y\|^2 + \cdots + \|Q^k y\|^2)^{1/2} \left\|(\lambda_1 Q^1 + \cdots + \lambda_k Q^k) y\right\|.$$

Cancelling the factor $(\|Q^1 y\|^2 + \cdots + \|Q^k y\|^2)^{1/2}$ in the above relation and using $Q^i y = Q^i x$, $\forall i$, yields the desired result. □

The following key lemma is essential to the proof of Theorem 3.1. It basically says that the norm of any convex combination of the derivatives of $f_1, \ldots, f_m$ is bounded uniformly from below over the zero set of these polynomials, provided that Assumption 3.1 holds.

LEMMA 3.5. *Suppose that Assumption* 3.1 *holds. Then there exists some constant* $\delta > 0$ *(depending on the problem data only) such that*

$$\left\|\sum_{i\in N} \lambda_i (Q^i x + b^i) + \sum_{i\in L} \lambda_i b^i\right\| \geq \delta \left(\sum_{i\in N} \lambda_i + \left\|\sum_{i\in L} \lambda_i b^i\right\|\right), \tag{3.6}$$

for all $\lambda \geq 0$ *and for all* x *with* $f_i(x) = 0$, $i = 1, \ldots, m$, *where* N, L *are the index sets for the nonlinear, linear constraints respectively.*

Proof. By translating x^* to the origin if necessary, we can assume that $x^* = 0$, in which case Assumption 3.1 implies $c_i > 0$ for $i \in N$ and $c_i \geq 0$ for $i \in L$. We shall prove (3.6) by induction on $|N|$, the cardinality of N. Clearly, (3.6) holds trivially when $|N| = 0$.

Assume the result is true for $|N| \leq k$; we show below that the result is also true for $|N| = k + 1$. Suppose the contrary. Then there must exist a sequence of $\{x^r\}$ and a sequence $\{\lambda^r\}$ such that

$$\left\|\sum_{i\in N} \lambda_i^r (Q^i x^r + b^i) + \sum_{i\in L} \lambda_i^r b^i\right\| \to 0, \tag{3.7}$$

where

$$\sum_{i\in N} \lambda_i^r + \left\|\sum_{i\in L} \lambda_i^r b^i\right\| = 1, \quad \lambda_i^r \geq 0, \quad f_i(x^r) = 0, \quad i = 1, \ldots, k+1. \tag{3.8}$$

Since $\|\sum_{i\in L} \lambda_i^r b^i\|$ is bounded, by Hoffman's error bound we may assume, without loss of generality, that $\{\lambda_i^r\}$, $i \in L$, is bounded. Note that $\{\lambda_i^r\}$, $i \in N$, is also bounded. Therefore, by passing onto a subsequence if necessary, we can assume that, for each $i \in N \cup L$, the sequence $\{\lambda_i^r\}$ converges to some limit, say, $\lambda_i^\infty \geq 0$. Let $N_1 \subseteq N$ denote the set of index i such that $\lambda_i^\infty > 0$, and let $\bar{N}_1$ denote the complement of N_1 in N. We claim that $N_1 \neq N$. Suppose the contrary, so that $\lambda_i^r \geq \theta$, for all r, for all $i \in N$, and for some $\theta > 0$. By (3.7)–(3.8), $\{\sum_{i\in N} \lambda_i^r Q^i x^r\}$ is bounded. Therefore, it follows from Lemma 3.4 that $\{\|Q^i x^r\|\}$, $i \in N$, is bounded. Since $f_i(x^r) = 0$, it follows that $\{b^i x^r\}$ is also bounded, for all i. Thus, by removing the part of x^r belonging to

the null space of Q^i and b^i if necessary, we may assume, without loss of generality, that $\{x^r\}$ is bounded. Let x^∞ be a cluster point of $\{x^r\}$. Then $f_i(x^\infty) = 0$ for all i and (cf. (3.7))

$$\sum_{i\in N} \lambda_i^\infty (Q^i x^\infty + b^i) + \sum_{i\in L} \lambda_i^\infty b^i = 0.$$

Let $f(x) = \sum_{i\in N\cup L} \lambda_i^\infty f_i(x)$. Then $f(x^\infty) = 0$ and $\nabla f(x^\infty) = 0$. Since each λ_i is nonnegative, $f(x)$ must be convex. Therefore, $f(x)$ attains its global minimum (in $\Re^n$), which is 0, at x^∞. However, since $\lambda_i > 0$ for all $i \in N$, Assumption 3.1 implies $f(x^*) < 0$, a contradiction. Thus, N_1 must be a proper subset of N.

We claim that

$$\|\lambda_i^r Q^i x^r\| \geq \epsilon \quad \forall r \geq r_0,\ i \in \bar{N}_1, \tag{3.9}$$

for some positive constant ϵ and some index r_0. In fact, if (3.9) does not hold, then there exists some index $j \in \bar{N}_1$ and a subsequence R such that $\{\|\lambda_j^r Q^j x^r\|\}_{r\in R}$ tends to 0. Hence (3.7) and the fact $\{\lambda_j^r\} \to 0$ imply that

$$\left\| \sum_{i\in N, i=j} \lambda_i^r (Q^i x^r + b^r) + \sum_{i\in L} \lambda_i^r b^i \right\| \to 0, \qquad r \in R,$$

and

$$\sum_{i\in N, i=j} \lambda_i^r + \left\| \sum_{i\in L} \lambda_i^r b^i \right\| \to 1.$$

The above two relations together with $f_i(x^r) = 0$, $\lambda_i^r \geq 0$ imply that (3.6) cannot hold for $N/\{j\}$ whose cardinality is k. This contradicts the inductive hypothesis. Hence (3.9) must hold.

For each $r \geq r_0$, consider the linear system in y

$$Q^i y = Q^i x^r, \quad \langle b^j, y\rangle = \langle b^j, x^r\rangle, \quad i \in N,\ j \in N_1 \cup L. \tag{3.10}$$

Clearly, x^r is a solution of the above linear system. Thus, by Hoffman's error bound (1.2), (3.10) has a solution y^r with the property

$$\|y^r\| \leq M \left(\sum_{i\in N} \|Q^i y^r\| + \sum_{i\in N_1\cup L} |\langle b^i, y^r\rangle| \right), \tag{3.11}$$

where M is some positive constant independent of r. By Lemma 3.3, there exists some $u > 0$ (depending on Q^i's only) such that

$$\langle Q^i y^r, y^r\rangle \geq u\|Q^i y^r\|^2 \quad \forall r.$$

Thus, we have

$$\begin{aligned} \lambda_i^r \langle Q^i y^r, y^r\rangle &\geq u\lambda_i^r \|Q^i y^r\|^2 = u\lambda_i^r \|Q^i x^r\|^2 \\ &\geq u\epsilon \|Q^i x^r\| = u\epsilon \|Q^i y^r\|, \qquad i \in \bar{N}_1,\ r \geq r_0, \end{aligned}$$

where the equalities follow from $Q^i x^r = Q^i y^r$ (cf. (3.10)) and the second inequality is due to (3.9). Since y^r satisfies (3.10), it follows that $f_i(y^r) = f_i(x^r) = 0$, for all $i \in N_1 \cup L$. Thus, we have

$$\langle b^i, y^r \rangle = c_i, \ \forall i \in L \quad \text{and} \quad |\langle b^i, y^r \rangle| \leq c_i + \langle y^r, Q^i y^r \rangle/2, \ i \in N_1.$$

Plugging the above three relations into (3.11) yields

$$\|y^r\| \leq M \left(\sum_{i \in \bar{N}_1} \frac{\lambda_i^r \langle Q^i y^r, y^r \rangle}{u\epsilon} + \sum_{i \in N_1} \frac{\sqrt{\langle Q^i y^r, y^r \rangle}}{\sqrt{u}} + \sum_{i \in N_1 \cup L} c_i + \sum_{i \in N_1} \frac{\langle y^r, Q^i y^r \rangle}{2} \right). \tag{3.12}$$

Also, it follows from $f_i(y^r) = 0$ that

$$\langle Q^i y^r + b^i, y^r \rangle = c_i + \langle y^r, Q^i y^r \rangle/2 > 0, \qquad i \in N_1, \tag{3.13}$$

where the inequality follows from $c_i > 0$ for $i \in N_1$ and the positive semidefiniteness of Q^i.

Using the relations (3.12)–(3.13) and (3.7), we will derive a contradiction. In particular, from (3.7) and $\{\lambda_i^r\} \to 0$ for $i \in \bar{N}_1$, it follows that

$$\left(\sum_{i \in N_1} \lambda_i^r (Q^i y^r + b^i) + \sum_{i \in \bar{N}_1} \lambda_i^r Q^i y^r + \sum_{i \in L} \lambda_i^r b^i \right) \to 0.$$

Notice that $\langle b^i, y^r \rangle = \langle b^i, x^r \rangle = c_i$, $i \in L$ (cf. (3.10)). So, computing the inner product of $y^r/\|y^r\|$ with the above expression yields

$$\sum_{i \in \bar{N}_1} \lambda_i^r \frac{\langle Q^i y^r, y^r \rangle}{\|y^r\|} + \sum_{i \in N_1} \lambda_i^r \frac{\langle Q^i y^r + b^i, y^r \rangle}{\|y^r\|} + \sum_{i \in L} \lambda_i^r \frac{c_i}{\|y^r\|} \to 0.$$

By the positive semidefiniteness of Q^i, every term in the first sum of the above expression is nonnegative, and, by (3.13) and $c_i \geq 0$, the terms in the last two sums are also nonnegative. Hence, the above relation implies (cf. $\lambda_i^\infty > 0$, $i \in N_1$),

$$\sum_{i \in \bar{N}_1} \lambda_i^r \frac{\langle Q^i y^r, y^r \rangle}{\|y^r\|} \to 0, \ \sum_{i \in N_1} \frac{c_i}{\|y^r\|} \to 0, \ \sum_{i \in N_1} \frac{\langle Q^i y^r, y^r \rangle}{\|y^r\|} \to 0, \ \sum_{i \in L} \lambda_i^r \frac{c_i}{\|y^r\|} \to 0. \tag{3.14}$$

We consider two cases.

Case 1. $N_1 = \emptyset$. Then $\bar{N}_1 = N$, in which case the first relation in (3.14) and (3.12) imply that $\{y^r\}$ is bounded. Since $Q^i x^r = Q^i y^r$ and $\lambda_i^r \to 0$ for $i \in \bar{N}_1$, it follows that $\lambda_i^r Q^i x^r \to 0$, contradicting (3.9).

Case 2. $N_1 \neq \emptyset$. Then the second relation in (3.14) implies $\|y^r\| \to \infty$, because $c_i > 0$ for $i \in N_1$. This, together with the third relation in (3.14), implies that

$$\sum_{i \in N_1} \frac{\sqrt{\langle Q^i y^r, y^r \rangle}}{\|y^r\|} \to 0. \tag{3.15}$$

Now, it can be seen that (3.14)–(3.15) contradict the relation (3.12). This completes the proof for the induction. □

Clearly, Lemma 3.5 implies the condition (3.3), so the error bound (3.4) holds for convex quadratic systems satisfying Assumption 3.1. This proves Theorem 3.1.

Compared to Hoffman's error bound (1.2) for a linear system, the error bound (3.2) has a weakness; namely, it only applies to convex quadratic *inequality* systems (cf. (3.1)). In general, the error bound fails to hold if equality constraints are present since the convexity may be lost. However, if the equality constraints are *linear* then the error bound (3.2) remains valid. In particular, consider the convex quadratic system

$$Ax = a, \qquad f_1(x) \leq 0, \quad f_2(x) \leq 0, \ldots, f_m(x) \leq 0.$$

Let P denote its solution set. Using Theorem 3.1, we can easily prove the following.

COROLLARY 3.6. *Suppose that there exists some $x^* \in P$ such that $f_i(y) < 0$ for all $i \in N$. Then there exists a constant $\tau > 0$ (depending on the problem data only) such that*

$$\operatorname{dist}(x, P) \leq \tau(\|[f(x)]_+\| + \|Ax - a\|) \quad \forall x \in \Re^n, \tag{3.16}$$

where $f(x) = (f_1(x), \ldots, f_m(x))^T$.

Error bound (3.16) strengthens the error bound (2.2) in two respects. First, (2.2) contains a factor $(1 + \|x\|)^{\gamma'}$ whereas (3.16) does not. Second, since the exponent γ in (2.2) is no larger than 1, the error bound (3.16) is sharper than (2.2) in general.

4. Discussions and possible extensions. In this paper, we have extended Hoffman's error bound to (possibly nonconvex) polynomial systems. Our results basically show that the solution set of a polynomial system is in general Hölder continuous. For the special case where the defining inequalities are convex quadratic and the equations are all linear, the corresponding solution set is Lipschitz continuous, provided that the system has a solution which satisfies the nonlinear inequalities strictly. This result sharpens an earlier error bound result by Mangasarian [Man85], which uses an additional asymptotic constraint qualification assumption.

It remains to be seen if and how the error bound results reported here can be applied to analyze the convergence and the rate of convergence of (non)linearly constrained convex optimization problems. So far, the second author has been able to apply the error bound (2.2) to the convergence analysis of a large class of primal-dual interior point algorithms for solving convex quadratic programs (see [Luo92]) without assuming strict complementarity.

Finally, we hope to know to what extent our error bound results can be generalized to nonpolynomial convex systems, without imposing any restrictive assumptions such as the boundedness of the solution set (see [Rob75]). Also, it would be interesting to see if there are systems other than the convex quadratic ones for which Hoffman's error bound hold without Mangasarian's asymptotic constraint qualification condition.

Acknowledgment. We are grateful to J.-S. Pang and Tao Wang of the Department of Mathematical Sciences, the Johns Hopkins University, who carefully read an early version of this paper and pointed out a gap in the proof of Lemma 3.5.

REFERENCES

[Hof52] A. J. HOFFMAN, *On approximate solutions of systems of linear inequalities,* J. Res. Natl. Bur. Standards, 49 (1952), pp. 263–265.

[Gor61] E. A. GORIN, *Asymptotic properties of polynomials and algebraic functions of several variables,* Uspehi Mat. Nauk (N.S.), 16, No. 1 (1961), pp. 93–119. (In Russian; English translation in Russian Mathematical Surveys.)

[Hor58] L. HÖRMANDER, *On the division of distributions by polynomials,* Ark. Mat., 3 (1958), pp. 555–568.

[Hor69] ———, *Linear Partial Differential Operators*, Springer-Verlag, New York, 1969.

[Luo92] Z.-Q. LUO, *Convergence Analysis of Primal-Dual Interior Point Algorithms for Solving Convex Quadratic Programs,* Communications Research Laboratory, McMaster University, Hamilton, Ontario, Canada, 1992.

[LuP93] Z.-Q. LUO AND J.-S. PANG, *Error Bounds for Analytic Systems and Their Applications,* Tech. Report 274, Communications Research Laboratory, McMaster University, Hamilton, Ontario, Canada, 1993; Department of Mathematical Sciences, The Johns Hopkins University, Baltimore, MD; Math. Programming, to appear.

[LuT92a] Z.-Q. LUO AND P. TSENG, *Error bound and convergence analysis of matrix splitting algorithms for the affine variational inequality problem,* SIAM J. Control Optim., 2 (1992), pp. 43–54.

[LuT92b] ———, *On the linear convergence of descent methods for convex essentially smooth minimization,* SIAM J. Control Optim., 30 (1992), pp. 408–425.

[MaD88] O. L. MANGASARIAN AND R. DE LEONE, *Error bounds for strongly convex programs and (super)linearly convergent iterative schemes for the least 2-norm solution of linear programs,* Appl. Math. Optim., 17 (1988), pp. 1–14.

[Man85] O. L. MANGASARIAN, *A condition number for differentiable convex inequalities,* Math. Oper. Res., 10 (1985), pp. 175–179.

[Pan87] J.-S. PANG, *A posteriori error bounds for the linearly-constrained variational inequality problem,* Math. Oper. Res., 12 (1987), pp. 474–484.

[Rob75] S. M. ROBINSON, *An application of error bounds for convex programming in a linear space,* SIAM J. Control Optim., 13 (1975), pp. 271–273.

SIAM J. OPTIMIZATION
Vol. 4, No. 2, pp. 393–422, May 1994
© 1994 Society for Industrial and Applied Mathematics
010

GLOBALLY CONVERGENT INEXACT NEWTON METHODS*

STANLEY C. EISENSTAT† AND HOMER F. WALKER‡

Abstract. Inexact Newton methods for finding a zero of $F : \mathbf{R}^n \to \mathbf{R}^n$ are variations of Newton's method in which each step only approximately satisfies the linear Newton equation but still reduces the norm of the local linear model of F. Here, inexact Newton methods are formulated that incorporate features designed to improve convergence from arbitrary starting points. For each method, a basic global convergence result is established to the effect that, under reasonable assumptions, if a sequence of iterates has a limit point at which F' is invertible, then that limit point is a solution and the sequence converges to it. When appropriate, it is shown that initial inexact Newton steps are taken near the solution, and so the convergence can ultimately be made as fast as desired, up to the rate of Newton's method, by forcing the initial linear residuals to be appropriately small. The primary goal is to introduce and analyze new inexact Newton methods, but consideration is also given to "globalizations" of (exact) Newton's method that can naturally be viewed as inexact Newton methods.

Key words. inexact Newton methods, Newton iterative methods, truncated Newton methods, Newton's method, globally convergent Newton-like methods, global convergence analyses for Newton-like methods

AMS subject classification. 65H10

1. Introduction.

Consider the system of nonlinear equations

$$F(x) = 0, \tag{1.1}$$

where $F : \mathbf{R}^n \to \mathbf{R}^n$ is continuously differentiable. To simplify the following discussion, we assume *only for the remainder of this introduction* that (1.1) has a solution x_* such that $F'(x_*)$ is invertible.

A classical algorithm for solving (1.1) is Newton's method.

ALGORITHM N (Newton's method).

LET x_0 BE GIVEN.
FOR $k = 0$ STEP 1 UNTIL "CONVERGENCE" DO:
  SOLVE $F'(x_k)\, s_k = -F(x_k)$.
  SET $x_{k+1} = x_k + s_k$.

The major strength of Newton's method lies in its local convergence properties: If x_0 is sufficiently close to x_*, then $\{x_k\}$ converges q-superlinearly to x_*. Usually, F' is Lipschitz continuous at x_*, and the convergence is q-quadratic. (See, e.g., Ortega and Rheinboldt [13, §10.2.2].)

At each stage of Newton's method, the Newton equation

$$F'(x_k)\, s_k = -F(x_k) \tag{1.2}$$

* Received by the editors March, 1991; accepted for publication (in revised form) March 29, 1993.
† Department of Computer Science and Research Center for Scientific Computation, Yale University, P. O. Box 2158, New Haven, Connecticut 06520. The work of this author was supported in part by Office of Naval Research contract N00014-86-K-0310 and National Science Foundation grant DCR-85-21451.
‡ Department of Mathematics and Statistics, Utah State University, Logan, Utah 84322–3900. The work of this author was supported in part by United States Department of Energy grants DE-FG02-86ER25018 and DE-FG02-92ER25136, National Science Foundation grant DMS-0088995, and United States Air Force Office of Scientific Research grant AFOSR-91-0294, all with Utah State University.

must be solved. Computing the exact solution can be expensive if n is large and, for any n, may not be justified when x_k is far from a solution. Thus, one might prefer to compute some approximate solution, leading to the following algorithm.

ALGORITHM IN (inexact Newton method [4]).

LET x_0 BE GIVEN.
FOR $k = 0$ STEP 1 UNTIL "CONVERGENCE" DO:
FIND **some** $\eta_k \in [0, 1)$ AND s_k THAT SATISFY

$$\|F(x_k) + F'(x_k)\, s_k\| \leq \eta_k \|F(x_k)\|. \tag{1.3}$$

SET $x_{k+1} = x_k + s_k$.

Since $F(x_k) + F'(x_k)\, s_k$ is the residual of (1.2), as well as the local linear model of F evaluated at s_k, each η_k reflects how accurately s_k solves (1.2); however, the method of determining s_k is not otherwise restricted.[1] We say that s_k satisfying (1.3) is an *inexact Newton step* (*at the level* η_k) and refer to (1.3) as an *inexact Newton condition*.

If x_0 is sufficiently close to x_* and the η_k's are uniformly bounded below one, then a sequence of inexact Newton iterates $\{x_k\}$ converges q-linearly to x_* in a well-chosen norm. If $\lim_{k\to\infty} \eta_k = 0$, then the convergence is q-superlinear. If F' is Lipschitz continuous at x_* and $\eta_k = O\|F(x_k)\|$, then the convergence is q-quadratic. In any case, the convergence can be made as fast as that of a sequence of exact Newton iterates by taking the η_k's to be appropriately small. (See Dembo, Eisenstat, and Steihaug [4].)

The convergence of Algorithms N and IN is only *local*, i.e., the iterates need not converge if x_0 is not near a solution. There are a number of "globalizations" of Algorithm N that help to improve the likelihood of convergence from arbitrary starting points. Most modern globalizations fit within the following framework at each iteration.

- Determine an initial trial step in some way related to the Newton step.
- Test the trial step to determine whether it gives adequate progress toward a solution; if it does not, then obtain a "more conservative" trial step according to the globalization strategy and repeat the test.

Trial steps are determined in a variety of ways but typically reduce the norm of the local linear model of F; thus these globalizations provide instances of Algorithm IN. In requiring adequate progress, a decrease in $\|F\|$ is imposed at each iteration that makes convergence to a solution likely. Nevertheless, the nature of the problem may still preclude convergence, and even if the iterates converge, the limit may be a local minimizer of $\|F\|$ that is not a solution, at which F' is necessarily singular.

The purpose of this paper is to introduce and analyze *globally convergent inexact Newton methods*. These are instances of Algorithm IN that fit within the above globalization framework for Algorithm N. All use a test for adequate progress that is based directly on the norms of F and its local linear model and is particularly compatible with the inexact Newton framework. This test can be viewed as an extension to the inexact Newton context of the type of tests considered by Moré and Sorensen [12] and Shultz, Schnabel, and Byrd [16] for unconstrained optimization and by Powell [14], El Hallabi [7], and El Hallabi and Tapia [8] for general nonlinear equations. It

[1] *Newton iterative* or *truncated Newton* methods, which use iterative methods to solve (1.2) approximately, constitute an important class of inexact Newton methods. Although these methods have provided considerable motivation, the developments in this paper are valid in the more general inexact Newton setting.

requires a decrease in $\|F\|$ at each iteration that makes convergence to a solution likely, although, as above, this may be precluded by the nature of the problem. For each algorithm, we establish, among other results, a basic global convergence result to the effect that, under reasonable assumptions, if a sequence of iterates has a limit point at which F' is invertible, then that limit point is a solution of (1.1) and the sequence converges to it. These results are counterparts for general nonlinear equations of results of Moré and Sorensen [12] and Shultz, Schnabel, and Byrd [16] for unconstrained optimization. When appropriate, we show in addition that initial inexact Newton trial steps are taken near a solution; therefore, as in the local theory of Dembo, Eisenstat, and Steihaug [4], the convergence can ultimately be made as fast as desired, up to the rate of Newton's method, by choosing the initial inexact Newton levels to be appropriately small.

Our primary goal is to provide a theoretical foundation for new inexact Newton algorithms. However, the analysis here also provides a general framework for treating globalizations of Algorithm N and other Newton-like algorithms, and we explore this in a number of applications. In §2, we introduce a general *global inexact Newton method*, on which all subsequent algorithms are based. In §3, we prove global convergence results for it. These developments primarily provide a foundation for §§5–8, but in §4 we give two applications to globalized Newton-like methods that show them to be of interest in their own right. In §5, we introduce and analyze two general algorithms; these serve as paradigms for more specific algorithms in §§6–8 that can be implemented as practical algorithms. The algorithms in §6 are *backtracking methods*: In these, if an initial inexact Newton step is unsatisfactory, then shorter steps in the same direction are taken until a satisfactory step is found. The algorithms in §7 are *equality curve methods*: In these, all trial steps satisfy specified inexact Newton conditions with equality; if an initial inexact Newton step is unsatisfactory, then steps at higher inexact Newton levels (which may vary in direction as well as length) are tried until a satisfactory step is found. In §8, we give two applications of the results in §7. In §9, we conclude with a summary discussion; the reader is encouraged to consult this at any time for a more detailed overview that indicates all algorithms in the paper, summarizes their properties, and shows their interrelationships.

For work that is similar in spirit to some of the work here, although differing in major details, see Steihaug [17, Chap. 5] and Brown and Saad [3, §3]. Less closely related, more specialized work can be found in Bank and Rose [2] and Deuflhard [6].

Preliminaries. *For the remainder of the paper, we assume only that* $F: \mathbf{R}^n \to \mathbf{R}^n$ *is continuously differentiable.* We make no assumptions about the invertibility of F' or about the existence of solutions of (1.1) unless otherwise noted.

The norm $\|\cdot\|$ is arbitrary, except where explicitly assumed to be an inner product norm or the Euclidean norm $\|\cdot\|_2$ in §§7 and 8. We denote $N_\delta(x) \equiv \{y \mid \|y-x\| < \delta\}$ for $\delta > 0$. We let $\arg\min_{x \in S} f(x)$ denote the set of arguments that minimize $f: \mathbf{R}^n \to \mathbf{R}^1$ over $S \subseteq \mathbf{R}^n$; if this set is a singleton, then we may treat it as a point.

We use the following technical results.

LEMMA 1.1 ([13, §2.3.3]). *Assume that* $F'(x)$ *is invertible. Then for any* $\epsilon > 0$, *there exists* $\delta > 0$ *such that* $F'(y)$ *is invertible and*

$$\|F'(y)^{-1} - F'(x)^{-1}\| < \epsilon$$

whenever $y \in N_\delta(x)$.

LEMMA 1.2 ([13, §3.2.10]). *For any* x *and* $\epsilon > 0$, *there exists* $\delta > 0$ *such that*

$$\|F(z) - F(y) - F'(y)(z-y)\| \le \epsilon \|z-y\|$$

whenever $y, z \in N_\delta(x)$.

In saying that x_* is a *limit point* of a sequence $\{x_k\}$, we mean that for every $\delta > 0$, there are infinitely many values of k for which $x_k \in N_\delta(x_*)$. Note that x_* is a limit point of $\{x_k\}$ if $x_k = x_*$ for infinitely many k.

In general, the existence of an inexact Newton step s_k satisfying (1.3) depends on $F(x_k)$, $F'(x_k)$, η_k, and the norm $\|\cdot\|$. There exists an inexact Newton step for *every* $\eta_k \in [0,1)$ if and only if $F(x) \in \text{range}\, F'(x_k)$, which always holds if $F'(x_k)$ is invertible. Recall that x is a *stationary point* of $\|F\|$ if $\|F(x)\| \leq \|F(x) + F'(x)\, s\|$ for every s. There is no inexact Newton step for *any* $\eta_k \in [0,1)$ if and only if $F(x_k) \neq 0$ and x_k is a stationary point of $\|F\|$; equivalently, there exists an inexact Newton step for *some* $\eta_k \in [0,1)$ if and only if either $F(x_k) = 0$ or x_k is not a stationary point of $\|F\|$. When $\|\cdot\|$ is an inner-product norm, there exists an inexact Newton step for *some* $\eta_k \in [0,1)$ if and only if either $F(x_k) = 0$ or $F(x_k) \not\perp \text{range}\, F'(x_k)$.

Many of the algorithms below are like Algorithm IN in calling for steps that satisfy inexact Newton conditions and perhaps other conditions but are not uniquely specified. We always assume that such suitable steps are found if they exist.

In saying that an algorithm *breaks down* at some x_k, we mean that it is somehow precluded from determining a suitable x_{k+1}. For example, Algorithm IN breaks down at some x_k if and only if there is no inexact Newton step from x_k, equivalently, if and only if $F(x_k) \neq 0$ and x_k is a stationary point of $\|F\|$. To discuss convergence in the following, we state algorithms so that they continue indefinitely if they do not break down. Thus, in saying that an algorithm *does not break down*, we mean simply that it generates an infinite sequence of iterates. In most cases below, we give conditions under which specific algorithms do not break down.

2. A global inexact Newton method. Our general global inexact Newton method below is obtained by augmenting the inexact Newton condition with a sufficient decrease condition on $\|F\|$. This provides the test for acceptability of an inexact Newton step that is used, in one form or another, in all of our algorithms.

ALGORITHM GIN (global inexact Newton method).

LET x_0 AND $t \in (0,1)$ BE GIVEN.
FOR $k = 0$ STEP 1 UNTIL ∞ DO:
FIND **some** $\eta_k \in [0,1)$ AND s_k THAT SATISFY

$$\|F(x_k) + F'(x_k)\, s_k\| \leq \eta_k \|F(x_k)\| \tag{2.1}$$

AND

$$\|F(x_k + s_k)\| \leq [1 - t(1 - \eta_k)]\, \|F(x_k)\|. \tag{2.2}$$

SET $x_{k+1} = x_k + s_k$.

Conditions (2.1) and (2.2) are closely related to certain tests for accepting a step considered elsewhere. For minimizing $f : \mathbf{R}^n \to \mathbf{R}^1$, the central feature of those tests is that, for a given $t \in (0,1)$, a step s_k from a current point x_k is acceptable if

$$ared_k(s_k) \geq t \cdot pred_k(s_k), \tag{2.3}$$

where $ared_k(s_k) \equiv f(x_k) - f(x_k + s_k)$ is the *actual reduction* and $pred_k(s_k)$ is the *predicted reduction* obtained from a local quadratic model of f (see Moré and Sorensen [12], Shultz, Schnabel, and Byrd [16], and the references therein). For general nonlinear equations, a special case of more general tests considered by Powell [14], El

Hallabi [7], and El Hallabi and Tapia [8] has the form (2.3), with

$$\begin{aligned} ared_k(s_k) &\equiv \|F(x_k)\| - \|F(x_k + s_k)\|, \\ pred_k(s_k) &\equiv \|F(x_k)\| - \|F(x_k) + F'(x_k)s_k\|, \end{aligned} \tag{2.4}$$

which we take as our definition of $ared_k(s_k)$ and $pred_k(s_k)$ here. With (2.4), conditions (2.1) and (2.2) can be rephrased as

$$pred_k(s_k) \geq (1 - \eta_k)\|F(x_k)\|, \tag{2.5}$$

$$ared_k(s_k) \geq t(1 - \eta_k)\|F(x_k)\|. \tag{2.6}$$

Note in particular that if s_k is any step that reduces the norm of the local linear model and if η_k is chosen so that (2.1)/(2.5) is satisfied with equality, then (2.6) is equivalent to (2.3). Thus (2.1) and (2.2) can be regarded as an extension of tests of the form (2.3) to the inexact Newton context.

Less closely related tests are the two *Goldstein–Armijo conditions* derived from work in [10] and [1]. Typically only the first (the "alpha condition") is implemented in practice (see, e.g., Dennis and Schnabel [5, Chap. 6]); this is

$$f(x_k + s_k) \leq f(x_k) + \alpha \nabla f(x_k)^T s_k, \tag{2.7}$$

where $f \equiv \frac{1}{2}\|F\|_2^2$ and $0 < \alpha < 1$. The following proposition shows that when $\|\cdot\| = \|\cdot\|_2$, the alpha condition implies (2.2) for steps satisfying (2.1). Consequently, algorithms that generate steps satisfying (2.1) and the alpha condition, such as those of Brown and Saad [3], can be regarded as special cases of Algorithm GIN.

PROPOSITION 2.1. *Let* $\|\cdot\| = \|\cdot\|_2$ *and* $f \equiv \frac{1}{2}\|F\|_2^2$. *If* (2.1) *and* (2.7) *hold, then* (2.2) *also holds with* $t = \alpha$.

Proof. It follows from (2.7) that

$$\|F(x_k + s_k)\|_2^2 \leq \|F(x_k)\|_2^2 + 2\alpha F(x_k)^T F'(x_k)\, s_k.$$

Writing $r \equiv F(x_k) + F'(x_k)\, s_k$, we obtain from (2.1) that

$$\begin{aligned} F(x_k)^T F'(x_k)\, s_k &= F(x_k)^T[-F(x_k) + r] \\ &= -\|F(x_k)\|_2^2 + F(x_k)^T r \\ &\leq -(1 - \eta_k)\|F(x_k)\|_2^2, \end{aligned}$$

whence

$$\|F(x_k + s_k)\|_2^2 \leq [1 - 2\alpha(1 - \eta_k)]\|F(x_k)\|_2^2.$$

Since the left-hand side is nonnegative, we must have $2\alpha(1 - \eta_k) \leq 1$; since $\sqrt{1 - \epsilon} \leq 1 - \epsilon/2$ whenever $|\epsilon| \leq 1$, we also have

$$\|F(x_k + s_k)\|_2 \leq [1 - \alpha(1 - \eta_k)]\, \|F(x_k)\|_2. \qquad \square$$

3. Convergence theorems. In this section, we develop a theoretical foundation for Algorithm GIN. This provides a basis for the analysis of all of the algorithms considered in the sequel. We first address breakdown in Lemma 3.1 and Corollary 3.2 below.

LEMMA 3.1. *Let x and $t \in (0,1)$ be given and assume that there exists an $\bar{s}$ that satisfies $\|F(x)+F'(x)\bar{s}\| < \|F(x)\|$. Then there exists $\eta_{\min} \in [0,1)$ such that, for any $\eta \in [\eta_{\min}, 1)$, there is an s satisfying*

$$\|F(x)+F'(x)s\| \le \eta\|F(x)\| \quad \textit{and} \quad \|F(x+s)\| \le [1-t(1-\eta)]\,\|F(x)\|.$$

Proof. Clearly $F(x) \neq 0$ and $\bar{s} \neq 0$. Set

$$\bar{\eta} \equiv \frac{\|F(x)+F'(x)\bar{s}\|}{\|F(x)\|},$$

$$\epsilon \equiv \frac{(1-t)(1-\bar{\eta})\|F(x)\|}{\|\bar{s}\|},$$

$$\eta_{\min} \equiv \max\left\{\bar{\eta},\ 1-\frac{(1-\bar{\eta})\delta}{\|\bar{s}\|}\right\},$$

where $\delta > 0$ is sufficiently small that

$$\|F(x+s)-F(x)-F'(x)s\| \le \epsilon\|s\|$$

whenever $\|s\| \le \delta$. Such a δ exists by Lemma 1.2.

For any $\eta \in [\eta_{\min}, 1)$, let $s \equiv \frac{1-\eta}{1-\bar{\eta}}\bar{s}$. Then

$$\begin{aligned}\|F(x)+F'(x)s\| &\le \frac{\eta-\bar{\eta}}{1-\bar{\eta}}\|F(x)\| + \frac{1-\eta}{1-\bar{\eta}}\|F(x)+F'(x)\bar{s}\| \\ &= \frac{\eta-\bar{\eta}}{1-\bar{\eta}}\|F(x)\| + \frac{1-\eta}{1-\bar{\eta}}\bar{\eta}\|F(x)\| \\ &= \eta\|F(x)\|,\end{aligned}$$

and, since

$$\|s\| = \frac{1-\eta}{1-\bar{\eta}}\|\bar{s}\| \le \frac{1-\eta_{\min}}{1-\bar{\eta}}\|\bar{s}\| \le \delta,$$

it follows that

$$\begin{aligned}\|F(x+s)\| &\le \|F(x+s)-F(x)-F'(x)s\| + \|F(x)+F'(x)s\| \\ &\le \epsilon\cdot\frac{1-\eta}{1-\bar{\eta}}\|\bar{s}\| + \eta\|F(x)\| \\ &= (1-t)(1-\eta)\,\|F(x)\| + \eta\|F(x)\| \\ &= [1-t(1-\eta)]\,\|F(x)\|. \qquad \square\end{aligned}$$

COROLLARY 3.2. *Algorithm* GIN *breaks down at some x_k if and only if there is no inexact Newton step from x_k, i.e., if and only if $F(x_k) \neq 0$ and x_k is a stationary point of $\|F\|$.*

The next theorem is used in the proof of the global convergence theorem that follows and is also of independent interest.

THEOREM 3.3. *Assume that $\{x_k\}$ is a sequence such that $F(x_k) \to 0$ and, for each k,*

$$\|F(x_k)+F'(x_k)s_k\| \le \eta\|F(x_k)\| \quad \textit{and} \quad \|F(x_{k+1})\| \le \|F(x_k)\|,$$

where $s_k \equiv x_{k+1} - x_k$ *and* η *is independent of* k. *If* x_* *is a limit point of* $\{x_k\}$ *such that* $F'(x_*)$ *is invertible, then* $F(x_*) = 0$ *and* $x_k \to x_*$.

Proof. Clearly $F(x_*) = 0$. Set $K \equiv \|F'(x_*)^{-1}\|$, and let $\delta > 0$ be sufficiently small that $F'(y)^{-1}$ exists and

$$\|F'(y)^{-1}\| \le 2K,$$

$$\|F(y) - F(x_*) - F'(x_*)(y - x_*)\| \le \frac{1}{2K}\|y - x_*\|,$$

whenever $y \in N_\delta(x_*)$. Such a δ exists by Lemmas 1.1 and 1.2.

If $y \in N_\delta(x_*)$, then

$$\begin{aligned}\|F(y)\| &\ge \|F'(x_*)(y - x_*)\| - \|F(y) - F(x_*) - F'(x_*)(y - x_*)\| \\ &\ge \frac{1}{\|F'(x_*)^{-1}\|}\|y - x_*\| - \frac{1}{2K}\|y - x_*\| \\ &= \frac{1}{2K}\|y - x_*\|,\end{aligned}$$

so that

$$\|y - x_*\| \le 2K\|F(y)\| \tag{3.1}$$

whenever $y \in N_\delta(x_*)$.

Let $\epsilon \in (0, \delta/4)$ be given. Since x_* is a limit point of $\{x_k\}$ and $F(x_*) = 0$, there is a k sufficiently large that

$$x_k \in S_\epsilon \equiv \{y \mid \|y - x_*\| < \delta/2 \text{ and } \|F(y)\| < \epsilon/[K(1+\eta)]\}.$$

Then

$$\begin{aligned}\|s_k\| &= \|F'(x_k)^{-1}\left[-F(x_k) + \{F(x_k) + F'(x_k)\, s_k\}\right]\| \\ &\le \|F'(x_k)^{-1}\|\big(\|F(x_k)\| + \|F(x_k) + F'(x_k)\, s_k\|\big) \\ &\le 2K(1+\eta)\|F(x_k)\| \\ &< 2\epsilon \\ &< \frac{\delta}{2},\end{aligned}$$

and so

$$\|x_{k+1} - x_*\| \le \|x_k - x_*\| + \|s_k\| < \delta.$$

Since

$$\|F(x_{k+1})\| \le \|F(x_k)\| < \epsilon/[K(1+\eta)]$$

and, by (3.1),

$$\|x_{k+1} - x_*\| \le 2K\|F(x_{k+1})\| < 2K \cdot \epsilon/[K(1+\eta)] < \delta/2,$$

it follows that $x_{k+1} \in S_\epsilon$. We conclude that $x_k \in S_\epsilon \subseteq N_\delta(x_*)$ for all sufficiently large k, and, since $\|F(x_k)\| \to 0$, it follows from (3.1) that $x_k \to x_*$. □

We now establish the basic global convergence theorem for Algorithm GIN.

THEOREM 3.4 (global convergence of algorithm GIN). *Assume that Algorithm* GIN *does not break down. If* $\sum_{k\geq 0}(1-\eta_k)$ *is divergent, then* $F(x_k) \to 0$. *If, in addition,* x_* *is a limit point of* $\{x_k\}$ *such that* $F'(x_*)$ *is invertible, then* $F(x_*) = 0$ *and* $x_k \to x_*$.

Proof. By (2.2),

$$\begin{aligned}\|F(x_k)\| &\leq [1-t(1-\eta_{k-1})]\,\|F(x_{k-1})\| \\ &\leq \|F(x_0)\| \prod_{0\leq j<k} [1-t(1-\eta_j)] \\ &\leq \|F(x_0)\| \exp\left[-t \sum_{0\leq j<k} (1-\eta_j)\right].\end{aligned}$$

Since $t > 0$ and $1-\eta_j \geq 0$, the divergence of $\sum_{k\geq 0}(1-\eta_k)$ implies $F(x_k) \to 0$. The remainder of the theorem follows from Theorem 3.3 with $\eta = 1$. □

With (2.2), the divergence of $\sum_{k\geq 0}(1-\eta_k)$ implies sufficient cumulative decrease in $\|F\|$ over all steps to ensure that $F(x_k) \to 0$. In applications of Theorem 3.4 in the sequel, conditions are given under which this divergence holds implicitly; it is not necessary to assume it explicitly. In general, however, it may not be possible to determine the η_k's so that $\sum_{k\geq 0}(1-\eta_k)$ is divergent, even though Algorithm GIN does not break down. For example, define $F(x) \equiv 1 + \exp(-x^2)$ for $x \in \mathbf{R}^1$. If $x_0 \neq 0$, then the full Newton step is always a well-defined inexact Newton step from the current x_k and, for any $t \in (0,1)$, admissible η_k and s_k exist by Lemma 3.1. However, $\sum_{k\geq 0}(1-\eta_k)$ cannot be divergent because $\{F(x_k)\}$ cannot converge to zero.

Note that (2.1) is not needed to obtain the weak result that $F(x_k) \to 0$; only (2.2) is used. However, (2.2) alone implies nothing further about the convergence of $\{x_k\}$. Indeed, for $F:\mathbf{R}^1 \to \mathbf{R}^1$ given by $F(x) \equiv x^2 - 1$, the solutions are ± 1; if $\{x_k\}$ converges to $+1$ and satisfies (2.2) for some $\{\eta_k\}$, then $\{(-1)^k x_k\}$ also satisfies (2.2) and has both $+1$ and -1 as limit points. To show that $x_k \to x_*$, we apply Theorem 3.3, which in effect uses (2.1) to ensure that iterates remain near solutions at which F' is invertible, thereby precluding the sort of behavior seen in this example.

Another way to state Theorem 3.4 is that if Algorithm GIN does not break down and $\sum_{k\geq 0}(1-\eta_k)$ is divergent, then $F(x_k) \to 0$ and one of the following holds:

1. $\|x_k\| \to \infty$, i.e., $\{x_k\}$ has no limit points;
2. $\{x_k\}$ has one or more limit points, and F' is singular at each of them;
3. $\{x_k\}$ converges to a solution x_* at which F' is invertible.

These alternatives allow useful corollaries to be drawn. For example, if F' is invertible on $\{x \mid \|F(x)\| \leq \|F(x_0)\|\}$ and this level set is bounded, then alternative 3 must hold and the ultimate rate of convergence is determined by the values of η_k for large k as in the local theory of Dembo, Eisenstat, and Steihaug [4].

Alternative 1 can occur. Indeed, if $F(x) \equiv \exp(-x^2)$ for $x \in \mathbf{R}^1$ and if $x_0 \neq 0$ and $t = \frac{1}{2}$, then full Newton steps are admissible in Algorithm GIN and $|x_k| \to \infty$.

Alternative 2 can also occur, and there can clearly be convergence to a point at which F' is singular. There can also be more than one limit point of $\{x_k\}$. For example, define $F(x) \equiv (x^{(1)}, 0)^T$ for $x = (x^{(1)}, x^{(2)})^T \in \mathbf{R}^2$. For each k, we take $\eta_k = \frac{1}{2}$ and $s_k = (-x_k^{(1)}/2, s_k^{(2)})^T$, where $x_k = (x_k^{(1)}, x_k^{(2)})^T$ and $s_k^{(2)}$ is arbitrary. Then (2.1) and (2.2) hold for any $t \in (0,1)$. Furthermore, the limit points of $\{x_k\}$ constitute

a subset of the solution set $\{(0, x^{(2)})^T\}$, and, since each $s_k^{(2)}$ is arbitrary, this subset can be made to contain any given subset of the solution set.

The next theorem is complementary to Theorem 3.3 and is useful in the sequel.

THEOREM 3.5. *Assume that Algorithm* GIN *does not break down. If x_* is a limit point of $\{x_k\}$ such that there exists a Γ independent of k for which*

$$\|s_k\| \le \Gamma(1-\eta_k)\|F(x_k)\| \tag{3.2}$$

whenever x_k is sufficiently near x_* and k is sufficiently large, then $x_k \to x_*$.

Proof. Suppose that $x_k \not\to x_*$. Let $\delta > 0$ be such that there exist infinitely many k for which $x_k \notin N_\delta(x_*)$ and sufficiently small that (3.2) holds whenever $x_k \in N_\delta(x_*)$ and k is sufficiently large.

Since x_* is a limit point of $\{x_k\}$, there exist $\{k_j\}$ and $\{\ell_j\}$ such that, for each j,

$$\begin{aligned} &x_{k_j} \in N_{\delta/j}(x_*), \\ &x_{k_j+i} \in N_\delta(x_*), \qquad i = 0, \dots, l_j - 1, \\ &x_{k_j+l_j} \notin N_\delta(x_*), \\ &k_j + l_j < k_{j+1}. \end{aligned}$$

Then for j sufficiently large,

$$\begin{aligned} \delta/2 &\le \|x_{k_j+l_j} - x_{k_j}\| \\ &\le \sum_{k=k_j}^{k_j+l_j-1} \|s_k\| \\ &\le \sum_{k=k_j}^{k_j+l_j-1} \Gamma(1-\eta_k)\|F(x_k)\| \\ &\le \sum_{k=k_j}^{k_j+l_j-1} \frac{\Gamma}{t}\{\|F(x_k)\| - \|F(x_{k+1})\|\} \\ &= \frac{\Gamma}{t}\{\|F(x_{k_j})\| - \|F(x_{k_j+l_j})\|\} \\ &\le \frac{\Gamma}{t}\{\|F(x_{k_j})\| - \|F(x_{k_{j+1}})\|\}. \end{aligned}$$

But the last right-hand side converges to zero since $x_{k_j} \to x_*$; hence, this inequality cannot hold for large j. □

Theorems 3.4 and 3.5 are still valid if we allow $\eta_k \in [0, 1]$ in Algorithm GIN. The resulting more general statements can be adapted to provide global convergence results for *any* sequence $\{x_k\}$ such that each $s_k \equiv x_{k+1} - x_k$ does not increase the norm of the local linear model of F, i.e., such that $pred_k(s_k) \ge 0$ for each k. Indeed, for each k, we can *define* $\eta_k \in [0, 1]$ by

$$\eta_k \equiv \begin{cases} \|F(x_k) + F'(x_k)\, s_k\|/\|F(x_k)\| & \text{if } F(x_k) \neq 0, \\ 0 & \text{otherwise.} \end{cases}$$

Then (2.1)/(2.5) holds with equality, and η_k and s_k are acceptable in Algorithm GIN if and only if (2.3) holds, i.e, $ared_k(s_k) \geq t \cdot pred_k(s_k)$. Moreover, introducing

$$relpred_k(s_k) \equiv \begin{cases} pred_k(s_k)/\|F(x_k)\| & \text{if } F(x_k) \neq 0, \\ 1 & \text{otherwise,} \end{cases}$$

we have

$$\sum_{k\geq 0}(1-\eta_k) = \sum_{k\geq 0} relpred_k(s_k).$$

Then Theorem 3.4 can be restated as follows.

COROLLARY 3.6. *Assume that* $\{x_k\}$ *is such that* $pred_k(s_k) \geq 0$ *and* $ared_k(s_k) \geq t \cdot pred_k(s_k)$ *for each* k, *where* $t \in (0,1)$ *is independent of* k. *If* $\sum_{k\geq 0} relpred_k(s_k)$ *is divergent, then* $F(x_k) \to 0$. *If, in addition,* x_* *is a limit point of* $\{x_k\}$ *such that* $F'(x_*)$ *is invertible, then* $F(x_*) = 0$ *and* $x_k \to x_*$.

This result can be viewed as an analogue of the results of Wolfe [18], [19] on sequences that proceed through descent-direction steps for $f:\mathbf{R}^n \to \mathbf{R}^1$ that satisfy the Goldstein–Armijo conditions. Theorem 3.5 can be restated in a similar manner.

COROLLARY 3.7. *Assume that* $\{x_k\}$ *is such that* $pred_k(s_k) \geq 0$ *and* $ared_k(s_k) \geq t \cdot pred_k(s_k)$ *for each* k, *where* $t \in (0,1)$ *is independent of* k. *If* x_* *is a limit point of* $\{x_k\}$ *such that there exists a* Γ *independent of* k *for which* $\|s_k\| \leq \Gamma \cdot pred_k(s_k)$ *whenever* x_k *is sufficiently near* x_* *and* k *is sufficiently large, then* $x_k \to x_*$.

4. Applications. Although the developments in §§2 and 3 are intended primarily to provide a groundwork for §§5–8, they also have broader applicability to globalizations of Algorithm N and related methods, which we now illustrate. Material in this section is not used in later sections.

Application 1. Trust region methods. There are a variety of trust region methods for unconstrained optimization and general nonlinear equations; see, e.g., Dennis and Schnabel [5, §6.4], El Hallabi [7], El Hallabi and Tapia [8], Moré and Sorensen [12], Shultz, Schnabel, and Byrd [16], and the references therein. Here, we show how the developments in §§2 and 3 can be brought to bear on these methods without giving an exhaustive treatment. The algorithm below is meant to reflect the essential features of several known trust region algorithms; it is especially close to Algorithm 4.2 of Moré and Sorensen [12].

ALGORITHM TR (trust region method).

LET x_0, $\bar{\delta}_0 > 0$, $0 < t \leq u < 1$, AND $0 < \theta_{\min} < \theta_{\max} < 1$ BE GIVEN.
FOR $k = 0$ STEP 1 UNTIL ∞ DO:
  SET $\delta_k = \bar{\delta}_k$ AND CHOOSE $s_k \in \arg\min_{\|s\|\leq\delta_k} \|F(x_k) + F'(x_k)\, s\|$.
  WHILE $ared_k(s_k) < t \cdot pred_k(s_k)$ DO:
    CHOOSE $\theta \in [\theta_{\min}, \theta_{\max}]$.
    UPDATE $\delta_k \leftarrow \theta\delta_k$ AND CHOOSE
      $s_k \in \arg\min_{\|s\|\leq\delta_k} \|F(x_k) + F'(x_k)\, s\|$.
  SET $x_{k+1} = x_k + s_k$.
  IF $ared_k(s_k) \geq u \cdot pred_k(s_k)$ CHOOSE $\bar{\delta}_{k+1} \geq \delta_k$;
    ELSE CHOOSE $\bar{\delta}_{k+1} \geq \theta_{\min}\delta_k$.

Algorithm TR and the analysis below could easily be modified to allow steps that are only approximate minimizers of the local linear model norm within trust regions, the use of a second norm in determining δ_k-balls as in El Hallabi [7] and El Hallabi

and Tapia [8], additional refinements in the determination of trust region radii, etc. It can easily be shown that Algorithm TR does not break down if, for each k, either $F'(x_k)$ is invertible or x_k is not a stationary point of $\|F\|$.

To develop a convergence analysis for Algorithm TR, we first use Corollary 3.7 to obtain the following lemma.

LEMMA 4.1. *Assume that Algorithm* TR *does not break down. Suppose that* x_* *is a limit point of* $\{x_k\}$ *such that there exists a* Γ *independent of* k *for which*

$$\|s_k\| \leq \Gamma \cdot pred_k(s_k) \tag{4.1}$$

whenever x_k *is sufficiently near* x_* *and* k *is sufficiently large. Then* $x_k \to x_*$ *and* $\liminf_{k\to\infty} \delta_k > 0$.

Remark. One can also conclude that x_* is a stationary point, but this is not needed here.

Proof. It is clear that $\{x_k\}$ satisfies the hypotheses of Corollary 3.7, and it follows immediately that $x_k \to x_*$. Choose $\delta > 0$ such that (4.1) holds whenever $x_k \in N_\delta(x_*)$ and k is sufficiently large and also such that

$$\|F(y) - F(x) - F'(x)\,(y - x)\| \leq \frac{(1-u)}{\Gamma}\|y - x\| \tag{4.2}$$

whenever $x, y \in N_\delta(x_*)$. Let k_0 be such that if $k \geq k_0$, then $x_k \in N_{\delta/2}(x_*)$ and (4.1) holds.

We claim that if $k \geq k_0$, then the while-loop in Algorithm TR terminates with

$$\delta_k \geq \min\left\{\bar{\delta}_{k_0}, \theta_{\min}\delta/2\right\},$$

whence $\liminf_{k\to\infty} \delta_k \geq \min\left\{\bar{\delta}_{k_0}, \theta_{\min}\delta/2\right\} > 0$. To show this, we first note that if $k \geq k_0$ and if s_k is a trial step for which $\|s_k\| \leq \delta/2$, then (4.2) and (4.1) give

$$\begin{aligned} ared_k(s_k) &\equiv \|F(x_k)\| - \|F(x_k + s_k)\| \\ &\geq \|F(x_k)\| - \|F(x_k) + F'(x_k)\,s_k\| - \|F(x_k + s_k) - F(x_k) - F'(x_k)\,s_k\| \\ &\geq pred_k(s_k) - \frac{(1-u)}{\Gamma}\|s_k\| \\ &\geq u \cdot pred_k(s_k). \end{aligned}$$

It follows from this that the while-loop terminates with

$$\delta_k \geq \min\left\{\bar{\delta}_k, \theta_{\min}\delta/2\right\}. \tag{4.3}$$

Indeed, if the while-loop does not reduce δ_k, then $\delta_k = \bar{\delta}_k$ on termination; whereas, if the while-loop reduces δ_k at least once, then the penultimate value is at least $\delta/2$, whence $\delta_k \geq \theta_{\min}\delta/2$ on termination. Furthermore, if $\delta_k \leq \delta/2$ on termination, then $\bar{\delta}_{k+1} \geq \delta_k$; whereas, if $\delta_k > \delta/2$ on termination, then $\bar{\delta}_{k+1} \geq \theta_{\min}\delta_k > \theta_{\min}\delta/2$. Thus

$$\bar{\delta}_{k+1} \geq \min\left\{\bar{\delta}_k, \theta_{\min}\delta/2\right\}. \tag{4.4}$$

The claim follows by induction from (4.3) and (4.4). □

The next two lemmas allow us to apply Lemma 4.1 in the proof of Theorem 4.4, which is our main result.

LEMMA 4.2. *If x_* is such that $F'(x_*)$ is invertible, then there exist Γ and $\epsilon_* > 0$ such that, for any $\delta > 0$,*

$$s \in \arg \min_{\|\bar{s}\| \le \delta} \|F(x) + F'(x)\,\bar{s}\| \tag{4.5}$$

satisfies

$$\|s\| \le \Gamma\{\|F(x)\| - \|F(x) + F'(x)\,s\|\} \tag{4.6}$$

whenever $x \in N_{\epsilon_*}(x_*)$.

Proof. Set $K \equiv \|F'(x_*)^{-1}\|$ and let $\epsilon_* > 0$ be sufficiently small that $F'(x)$ is invertible and $\|F'(x)^{-1}\| \le 2K$ whenever $x \in N_{\epsilon_*}(x_*)$. Suppose that $x \in N_{\epsilon_*}(x_*)$ and s is given by (4.5) for an arbitrary $\delta > 0$. Denote $s^{\mathrm{N}} \equiv -F'(x)^{-1}F(x)$, and note that $\|s\| \le \|s^{\mathrm{N}}\|$ since s^{N} is the unique global minimizer of the norm of the local linear model. If $s^{\mathrm{N}} = 0$, then $s = 0$ and (4.6) holds trivially for any Γ. If $s^{\mathrm{N}} \ne 0$, then

$$\begin{aligned}
\|F(x)\| - \|F(x) + F'(x)\,s\| &\ge \|F(x)\| - \left\|F(x) + F'(x)\,\frac{\|s\|}{\|s^{\mathrm{N}}\|}s^{\mathrm{N}}\right\| \\
&= \frac{\|s\|}{\|s^{\mathrm{N}}\|}\|F(x)\| \\
&\ge \frac{\|s\|}{2K},
\end{aligned}$$

and (4.6) holds with $\Gamma \equiv 2K$. Thus (4.6) holds with $\Gamma \equiv 2K$ $\forall x \in N_{\epsilon_*}(x_*)$. □

LEMMA 4.3. *If x_* is not a stationary point of $\|F\|$, then there exist Γ, $\delta_* > 0$, and $\epsilon_* > 0$ such that s given by (4.5) satisfies (4.6) whenever $x \in N_{\epsilon_*}(x_*)$ and $0 < \delta \le \delta_*$.*

Proof. Let $\epsilon > 0$ be such that if $x \in N_\epsilon(x_*)$, then $\|F(x)\| \ge \frac{1}{2}\|F(x_*)\|$. Let s_* be such that $\|F(x_*) + F'(x_*)\,s_*\| < \|F(x_*)\|$. Choose η_* such that

$$\|F(x_*) + F'(x_*)\,s_*\|/\|F(x_*)\| < \eta_* < 1.$$

Since F and F' are continuous, there exists $\epsilon_* \in (0, \epsilon]$ such that

$$\|F(x) + F'(x)\,s_*\| \le \eta_*\|F(x)\|$$

whenever $x \in N_{\epsilon_*}(x_*)$. Choose $\delta_* \in (0, \|s_*\|)$. Suppose that $x \in N_{\epsilon_*}(x_*)$ and $0 < \delta \le \delta_*$. For s given by (4.5), we have

$$\begin{aligned}
\|F(x)\| - \|F(x) + F'(x)\,s\| &\ge \|F(x)\| - \left\|F(x) + F'(x)\,\frac{\|s\|}{\|s_*\|}s_*\right\| \\
&\ge \|F(x)\| - \left(1 - \frac{\|s\|}{\|s_*\|}\right)\|F(x)\| \\
&\qquad - \frac{\|s\|}{\|s_*\|}\|F(x) + F'(x)\,s_*\| \\
&\ge \frac{(1-\eta_*)\|F(x)\|}{\|s_*\|}\|s\| \\
&\ge \frac{(1-\eta_*)\|F(x_*)\|}{2\|s_*\|}\|s\|,
\end{aligned}$$

and (4.6) holds with $\Gamma \equiv 2\|s_*\|/[(1-\eta_*)\|F(x_*)\|]$. □

THEOREM 4.4. *Assume that Algorithm* TR *does not break down. Then every limit point of $\{x_k\}$ is a stationary point of $\|F\|$. If x_* is a limit point of $\{x_k\}$ such that $F'(x_*)$ is invertible, then $F(x_*) = 0$ and $x_k \to x_*$; furthermore, $s_k = -F'(x_k)^{-1}F(x_k)$, the full Newton step, whenever k is sufficiently large.*

Proof. Suppose that x_* is a limit point of $\{x_k\}$ that is not a stationary point of $\|F\|$.

We claim that, for any $\delta > 0$, there exists an $\epsilon > 0$ such that if $x_k \in N_\epsilon(x_*)$ and k is sufficiently large, then $\delta_k \le \delta$. Otherwise, there would exist a $\delta > 0$ and $\{x_{k_j}\} \subseteq \{x_k\}$ such that $x_{k_j} \to x_*$ and $\delta_{k_j} > \delta$ for each j. Then

$$
\begin{aligned}
0 &= \lim_{j\to\infty} \{\|F(x_{k_j})\| - \|F(x_{k_j+1})\|\} \\
&\ge \liminf_{j\to\infty} \{\|F(x_{k_j})\| - \|F(x_{k_j+1})\|\} \\
&= \liminf_{j\to\infty} ared_{k_j}(s_{k_j}) \\
&\ge t \cdot \liminf_{j\to\infty} pred_{k_j}(s_{k_j}) \\
&= t \cdot \liminf_{j\to\infty} \Big\{ \|F(x_{k_j})\| - \min_{\|s\|\le\delta_{k_j}} \|F(x_{k_j}) + F'(x_{k_j})\, s\| \Big\} \\
&\ge t \cdot \lim_{j\to\infty} \Big\{ \|F(x_{k_j})\| - \min_{\|s\|\le\delta} \|F(x_{k_j}) + F'(x_{k_j})\, s\| \Big\} \\
&= t \cdot \Big\{ \|F(x_*)\| - \min_{\|s\|\le\delta} \|F(x_*) + F'(x_*)\, s\| \Big\}.
\end{aligned}
$$

But the last right-hand side must be positive since x_* is not a stationary point.

Now let Γ, δ_*, and ϵ_* be as in Lemma 4.3. By the above claim, there exists $\epsilon \in (0, \epsilon_*]$ such that if $x_k \in N_\epsilon(x_*)$ and k is sufficiently large, then $\delta_k \le \delta_*$. By Lemma 4.3, (4.1) holds for Γ independent of k whenever $x_k \in N_\epsilon(x_*)$ and k is sufficiently large. Then Lemma 4.1 implies that $x_k \to x_*$ and $\liminf_{k\to\infty} \delta_k > 0$. But since the claim implies that $\delta_k \to 0$, this is a contradiction. Hence, x_* must be a stationary point.

Suppose that x_* is a limit point of $\{x_k\}$ such that $F'(x_*)$ is invertible. Since x_* must be a stationary point, we must have $F(x_*) = 0$. It follows from Lemma 4.2 that there exists a Γ independent of k for which (4.1) holds whenever x_k is sufficiently near x_*. Then Lemma 4.1 implies that $x_k \to x_*$ and there exists a $\delta > 0$ such that $\delta_k \ge \delta$ for sufficiently large k. Since $x_k \to x_*$ and $F(x_k) \to F(x_*) = 0$, we have that $\|F'(x_k)^{-1}F(x_k)\| \le \delta \le \delta_k$ and, therefore, $s_k = -F'(x_k)^{-1}F(x_k)$ whenever k is sufficiently large. □

Application 2. Global approximate Newton methods. The global approximate Newton method of Bank and Rose [2] can be written as follows.

ALGORITHM GAN (global approximate Newton method [2]).

LET x_0 BE GIVEN; DETERMINE $K_0 \ge 0$.
FOR $k = 0$ STEP 1 UNTIL ∞ DO:
  SOLVE $M_k \bar{s}_k = -F(x_k)$, WHERE $M_k \approx F'(x_k)$.
  CHOOSE $K_k \in [0, K_0]$.
  SET $s_k = \tau_k \bar{s}_k$, WHERE $\tau_k \equiv 1/(1 + K_k\|F(x_k)\|)$.
  SET $x_{k+1} = x_k + s_k$.

Here, we show that Algorithm GAN is a special case of Algorithm GIN and that the global convergence result below of Bank and Rose [2, §2] follows from Theorem 3.4 under the assumptions in [2, §2], viz., the following:

(1) $L(x_0) \equiv \{x|\ \|F(x)\| \leq \|F(x_0)\|\}$ is bounded;
(2) F' is invertible on $L(x_0)$, each M_k is invertible, and $\|M_k^{-1}\| \leq \kappa$ for all $k \geq 0$;
(3) $\|F'(y) - F'(x)\| \leq \gamma\|y - x\|$ for $x, y \in \{u|\ \|u\| \leq \sup_{v \in L(x_0)} \|v\| + \kappa\|F(x_0)\|\}$;
(4) $F(x_k) \neq 0$ and $\bar{\eta}_k \equiv \|F(x_k) + F'(x_k)\bar{s}_k\|/\|F(x_k)\| \leq \bar{\eta}_0 < 1$ for all $k \geq 0$;
(5) For $t \in (0, 1 - \bar{\eta}_0)$, $K_k \geq (\kappa^2\gamma/2)(1 - \bar{\eta}_k - t)^{-1} - \|F(x_k)\|^{-1}$ for all $k \geq 0$.

PROPOSITION 4.5 (cf. conclusion (i) of [2, Theorem 1, p. 285]). *Under assumptions* (1)–(5) *above, there exists an* x_* *such that* $F(x_*) = 0$ *and* $x_k \to x_*$.

Proof. Setting $\eta_k \equiv (1 - \tau_k) + \tau_k\bar{\eta}_k$ for each k, we have that $\eta_k \in [0, 1)$ and

$$\begin{aligned}\|F(x_k) + F'(x_k)s_k\| &= \|(1 - \tau_k)F(x_k) + \tau_k[F(x_k) + F'(x_k)\bar{s}_k]\| \\ &\leq \eta_k\|F(x_k)\|,\end{aligned}$$

i.e., (2.1) holds. By Bank and Rose [2, (2.18), p. 283], we also have

$$\|F(x_k + s_k)\| \leq (1 - t\tau_k)\|F(x_k)\|.$$

Since

$$1 - t\tau_k \leq 1 - t\tau_k(1 - \bar{\eta}_k) = 1 - t(1 - \eta_k),$$

this gives

$$\|F(x_k + s_k)\| \leq [1 - t(1 - \eta_k)]\,\|F(x_k)\|,$$

i.e., (2.2) holds. Thus Algorithm GAN is a special case of Algorithm GIN.

To complete the proof, we note that

$$1 - \eta_k = \tau_k(1 - \bar{\eta}_k) \geq \tau_k(1 - \bar{\eta}_0)$$

and

$$\tau_k \equiv \frac{1}{1 + K_k\|F(x_k)\|} \geq \frac{1}{1 + K_0\|F(x_0)\|};$$

hence, $\sum_{k \geq 0}(1 - \eta_k)$ is divergent. Moreover, assumptions (1) and (2) imply that $\{x_k\}$ has a limit point x_* such that $F'(x_*)$ is invertible. Then Theorem 3.4 implies that $F(x_*) = 0$ and $x_k \to x_*$. □

5. Two paradigm methods. We begin now to develop methods that test initial inexact Newton steps and, if they are unsatisfactory, determine new steps as necessary until steps satisfying (2.1) and (2.2) are found. Our ultimate goal is to outline particular algorithms in §§6–8 that can be implemented as practical algorithms. To simplify their treatment, we formulate and analyze in this section two paradigm methods that incorporate their basic features.

At each iteration of these paradigm methods, an initial inexact Newton step at a specified level is tried and, if it proves unsatisfactory, then inexact Newton steps at higher levels are tried until a satisfactory step is found. Accordingly, we assume that, at the kth iteration of each method, there is a means of specifying *some* inexact

Newton step from x_k for each η in $[\bar{\eta}_k, 1]$, where $\bar{\eta}_k \in [0, 1)$ is the initial inexact Newton level. We express this by assuming that, for each k, there is a curve σ_k satisfying

$$\|F(x_k) + F'(x_k)\,\sigma_k(\eta)\| \le \eta \|F(x_k)\|, \qquad \bar{\eta}_k \le \eta \le 1. \tag{5.1}$$

For now, σ_k is not required to have any properties other than (5.1).

We call the first method a *minimum reduction method*: At each iteration, the initial inexact Newton step is required to achieve at least a prescribed minimum reduction of the norm of the local linear model.

ALGORITHM MR (minimum reduction method).

LET x_0, $\eta_{\max} \in [0, 1)$, $t \in (0, 1)$, AND $0 < \theta_{\min} < \theta_{\max} < 1$ BE GIVEN.
FOR $k = 0$ STEP 1 UNTIL ∞ DO:
  CHOOSE $\bar{\eta}_k \in [0, \eta_{\max}]$.
  DETERMINE σ_k SUCH THAT (5.1) HOLDS; SET $\eta_k = \bar{\eta}_k$.
  WHILE $\|F(x_k + \sigma_k(\eta_k))\| > [1 - t(1 - \eta_k)]\,\|F(x_k)\|$ DO:
    CHOOSE $\theta \in [\theta_{\min}, \theta_{\max}]$.
    UPDATE $\eta_k \leftarrow 1 - \theta(1 - \eta_k)$.
  SET $x_{k+1} = x_k + \sigma_k(\eta_k)$.

We call the second method a *trust level method*: At each iteration, the initial reduction required in the local linear model norm reflects a level of trust in the model based on behavior observed at previous iterations. The term "trust level" is intended both to evoke and to contrast this method with trust *region* methods, in which steps are determined within *regions* of trust of local linear models of F. The structure of this method has obvious parallels with that of Algorithm TR in §4.

ALGORITHM TL (trust level method).

LET x_0, $\bar{\eta}_0 \in [0, 1)$, $0 < t \le u < 1$, AND $0 < \theta_{\min} < \theta_{\max} < 1$ BE GIVEN.
FOR $k = 0$ STEP 1 UNTIL ∞ DO:
  DETERMINE σ_k SUCH THAT (5.1) HOLDS; SET $\eta_k = \bar{\eta}_k$.
  WHILE $\|F(x_k + \sigma_k(\eta_k))\| > [1 - t(1 - \eta_k)]\,\|F(x_k)\|$ DO:
    CHOOSE $\theta \in [\theta_{\min}, \theta_{\max}]$.
    UPDATE $\eta_k \leftarrow 1 - \theta(1 - \eta_k)$.
  SET $x_{k+1} = x_k + \sigma_k(\eta_k)$.
  IF $\|F(x_{k+1})\| \le [1 - u(1 - \eta_k)]\,\|F(x_k)\|$ CHOOSE $\bar{\eta}_{k+1} \in [0, \eta_k]$;
    ELSE CHOOSE $\bar{\eta}_{k+1} \in [0, 1 - \theta_{\min}(1 - \eta_k)]$.

The restriction $\bar{\eta}_k \in [0, \eta_{\max}]$ in Algorithm MR should not be a serious practical restriction since $\eta_{\max}$ can be taken arbitrarily near one. The restriction $\theta \in [\theta_{\min}, \theta_{\max}]$ in the while-loop in both algorithms is reflective of typical "safeguarding" practices; see, e.g., Dennis and Schnabel [5, §§6.3.2 and 6.4.3]. This still allows a great deal of flexibility in the choice of θ.

Whether either algorithm breaks down at the kth iteration depends on both the existence and the nature of σ_k satisfying (5.1). *Some* such σ_k exists if and only if there exists *some* inexact Newton step at the level $\bar{\eta}_k$. Indeed, the existence of such an inexact Newton step is clearly necessary, and if $\bar{s}_k$ is such an inexact Newton step, then the backtracking curve

$$\sigma_k(\eta) \equiv \frac{1 - \eta}{1 - \bar{\eta}_k}\bar{s}_k, \qquad \bar{\eta}_k \le \eta \le 1, \tag{5.2}$$

satisfies (5.1). The lemma below gives useful conditions under which neither algorithm breaks down in the while-loop once σ_k satisfying (5.1) has been found.

LEMMA 5.1. *At the kth step of Algorithm* MR (*Algorithm* TL), *if* $F(x_k) \neq 0$ *and there exists a* Γ *for which*

$$\|\sigma_k(\eta)\| \leq \Gamma(1-\eta)\|F(x_k)\|, \qquad \bar{\eta}_k \leq \eta \leq 1, \tag{5.3}$$

then the while-loop terminates with

$$1-\eta_k \geq \min\left\{1-\bar{\eta}_k, \frac{\theta_{\min}\delta}{\Gamma\|F(x_k)\|}\right\} \tag{5.4}$$

for any $\delta > 0$ *sufficiently small that*

$$\|F(x) - F(x_k) - F'(x_k)(x-x_k)\| \leq \frac{1-t}{\Gamma}\|x-x_k\| \tag{5.5}$$

whenever $x \in N_\delta(x_k)$.

Remark. If σ_k is given by (5.2) and $F(x_k) \neq 0$, then (5.3) holds with $\Gamma \equiv \|\bar{s}_k\|/[(1-\bar{\eta}_k)\|F(x_k)\|]$. It follows from Lemma 5.1 and the preceding discussion that, for either algorithm, if $F(x_k) \neq 0$, then there exists *some* σ_k for which breakdown is avoided at the kth step if and only if there exists *some* inexact Newton step at the level $\bar{\eta}_k$.

Proof. Note that if $\eta \in [\bar{\eta}_k, 1]$ is such that $1-\eta < \delta/(\Gamma\|F(x_k)\|)$, then (5.3), (5.1), and (5.5) imply that $\|\sigma_k(\eta)\| < \delta$ and

$$\begin{aligned}
\|F(x_k+\sigma_k(\eta))\| &\leq \|F(x_k) + F'(x_k)\,\sigma_k(\eta)\| \\
&\qquad + \|F(x_k+\sigma_k(\eta)) - F(x_k) - F'(x_k)\,\sigma_k(\eta)\| \\
&\leq \eta\|F(x_k)\| + \frac{1-t}{\Gamma}\|\sigma_k(\eta)\|, \\
&\leq [1-t(1-\eta)]\|F(x_k)\|.
\end{aligned}$$

Since $1-\eta_k$ is reduced by a factor $\theta \leq \theta_{\max} < 1$ at each iteration of the while-loop, it follows that the while-loop terminates.

Suppose that η_k is the final value determined by the while-loop. If $\eta_k = \bar{\eta}_k$, then (5.4) is immediate, so suppose that $\eta_k \neq \bar{\eta}_k$, i.e., the body of the while-loop has been executed at least once. Denoting the penultimate value by η_k^-, we see from the observation above that $1-\eta_k^- \geq \delta/(\Gamma\|F(x_k)\|)$. Since $1-\eta_k = \theta(1-\eta_k^-)$ for some $\theta \geq \theta_{\min}$, it follows that $1-\eta_k \geq \theta_{\min}\delta/(\Gamma\|F(x_k)\|)$. □

Algorithms MR and TL are clearly special cases of Algorithm GIN, and we use Theorems 3.4 and 3.5 to obtain the global convergence result below.

THEOREM 5.2 (global convergence of algorithms MR and TL). *Assume that Algorithm* MR (*Algorithm* TL) *does not break down. If* x_* *is a limit point of* $\{x_k\}$ *such that there exists a* Γ *independent of* k *for which*

$$\|\sigma_k(\eta)\| \leq \Gamma(1-\eta)\|F(x_k)\|, \qquad \bar{\eta}_k \leq \eta \leq 1, \tag{5.6}$$

holds whenever x_k *is sufficiently near* x_*, *then* $F(x_*) = 0$ *and* $x_k \to x_*$. *Furthermore,* $\eta_k = \bar{\eta}_k$ *for all sufficiently large* k.

Remark. If $\{x_k\}$ generated by Algorithm MR or Algorithm TL converges to a solution at which F' is invertible and if $\eta_k = \bar{\eta}_k$ for all sufficiently large k, then the

ultimate rate of convergence is governed by the choices of the $\bar{\eta}_k$'s as in the local theory of Dembo, Eisenstat, and Steihaug [4]. Analogous statements hold for the results in §§6–8 that are derived from Theorem 5.2.

Proof. It follows immediately from Theorem 3.5 that $x_k \to x_*$.

If $F(x_k) = 0$ for some k, then $F(x_j) = 0$ for all $j \geq k$ and the remainder of the theorem follows easily. Suppose that $F(x_k) \neq 0$ for every k. Let $\delta > 0$ be sufficiently small that (5.6) holds whenever $x_k \in N_\delta(x_*)$ and also

$$\|F(y) - F(x) - F'(x)\,(y - x)\| \leq \frac{1-t}{\Gamma}\,\|y - x\|$$

whenever x, $y \in N_{2\delta}(x_*)$. It follows from Lemma 5.1 that (5.4) holds on termination of the while-loop whenever $x_k \in N_\delta(x_*)$. Set $K \equiv \sup_{x \in N_\delta(x_*)} \|F(x)\|$.

To apply Theorem 3.4, we show that $\sum_{k \geq 0}(1 - \eta_k)$ is divergent. In the case of Algorithm MR, it follows from (5.4) that whenever $x_k \in N_\delta(x_*)$, we have

$$1 - \eta_k \geq \min\left\{1 - \eta_{\max}, \frac{\theta_{\min}\delta}{\Gamma K}\right\}$$

on termination of the while-loop. Since $x_k \to x_*$, there are infinitely many k for which $x_k \in N_\delta(x_*)$; therefore, $\sum_{k \geq 0}(1 - \eta_k)$ is divergent in this case. In the case of Algorithm TL, we set $\epsilon \equiv (1-u)/\Gamma$ and take δ smaller if necessary so that

$$\|F(y) - F(x) - F'(x)\,(y - x)\| \leq \epsilon \|y - x\| \tag{5.7}$$

whenever x, $y \in N_\delta(x_*)$. Suppose that k_0 is sufficiently large that $x_k \in N_\delta(x_*)$ whenever $k \geq k_0$. Then for $k \geq k_0$, (5.7) and (5.6) give

$$\begin{aligned}
\|F(x_{k+1})\| &\leq \|F(x_k) + F'(x_k)\,\sigma_k(\eta_k)\| \\
&\qquad + \|F(x_{k+1}) - F(x_k) - F'(x_k)\,\sigma_k(\eta_k)\| \\
&\leq \eta_k \|F(x_k)\| + \epsilon \|\sigma_k(\eta_k)\| \\
&\leq [1 - u(1 - \eta_k)]\|F(x_k)\|,
\end{aligned}$$

which implies $\bar{\eta}_{k+1} \leq \eta_k$. Then (5.4) and an easy induction give

$$\begin{aligned}
1 - \eta_{k+1} &\geq \min\left\{1 - \bar{\eta}_{k+1}, \frac{\theta_{\min}\delta}{\Gamma K}\right\} \\
&\geq \min\left\{1 - \eta_k, \frac{\theta_{\min}\delta}{\Gamma K}\right\} \\
&\geq \min\left\{1 - \eta_{k_0}, \frac{\theta_{\min}\delta}{\Gamma K}\right\}.
\end{aligned}$$

It follows that $\sum_{k \geq 0}(1 - \eta_k)$ is divergent in this case as well.

Since $\sum_{k \geq 0}(1 - \eta_k)$ is divergent, Theorem 3.4 implies that $F(x_*) = 0$. Then $F(x_k) \to 0$, and it follows from (5.4) that $\eta_k = \bar{\eta}_k$ whenever k is sufficiently large. □

6. Backtracking methods. In this section, we outline backtracking algorithms that can be easily implemented as practical algorithms once a means of determining initial inexact Newton steps has been provided. These algorithms are based on Algorithm MR in §5; Algorithm TL does not provide a useful paradigm, as noted below. Developments in this section are not used in later sections.

The principal algorithm in this section is the following.

ALGORITHM INB (inexact Newton backtracking method).

LET x_0, $\eta_{\max} \in [0,1)$, $t \in (0,1)$, AND $0 < \theta_{\min} < \theta_{\max} < 1$ BE GIVEN.
FOR $k = 0$ STEP 1 UNTIL ∞ DO:
  CHOOSE $\bar{\eta}_k \in [0, \eta_{\max}]$ AND DETERMINE $\bar{s}_k$ SUCH THAT
    $\|F(x_k) + F'(x_k)\,\bar{s}_k\| \le \bar{\eta}_k \|F(x_k)\|$.
  SET $s_k = \bar{s}_k$ AND $\eta_k = \bar{\eta}_k$.
  WHILE $\|F(x_k + s_k)\| > [1 - t(1 - \eta_k)]\,\|F(x_k)\|$ DO:
    CHOOSE $\theta \in [\theta_{\min}, \theta_{\max}]$.
    UPDATE $s_k \leftarrow \theta s_k$ AND $\eta_k \leftarrow 1 - \theta(1 - \eta_k)$.
  SET $x_{k+1} = x_k + s_k$.

Algorithm INB is a special case of Algorithm MR. Indeed, for each k, if σ_k is the backtracking curve given previously by (5.2), i.e.,

$$\sigma_k(\eta) \equiv \frac{1-\eta}{1-\bar{\eta}_k}\bar{s}_k, \qquad \bar{\eta}_k \le \eta \le 1, \tag{6.1}$$

then σ_k satisfies (5.1) and an easy induction shows that $s_k = \sigma_k(\eta_k)$ in the while-loop of Algorithm INB. Algorithm MR provides a useful paradigm here because if σ_k is given by (6.1) with $\bar{\eta}_k \in [0, \eta_{\max}]$, then we can show that (5.3) holds for Γ independent of k near a point at which F' is invertible and we can apply Theorem 5.2. Algorithm TL does not provide a useful paradigm because no inequality of the form (5.3) can hold for *all* backtracking curves if $\bar{\eta}_k$ is not bounded away from one.

Algorithm INB does not break down at the kth step if a suitable $\bar{s}_k$ can be found and if either $F(x_k) \neq 0$ or $F'(x_k)$ is invertible. Indeed, suppose we have found $\bar{s}_k$. If $F(x_k) \neq 0$, then it follows from Lemma 5.1 and the subsequent remark that the algorithm does not break down in the while-loop; if $F(x_k) = 0$ and $F'(x_k)$ is invertible, then $\bar{s}_k = 0$ and the while-loop terminates immediately. Note that if $F'(x_k)$ is invertible, then a suitable $\bar{s}_k$ exists and the algorithm does not break down at the kth step. Similar remarks hold for the other algorithms outlined in this section.

THEOREM 6.1 (global convergence of algorithm INB). *Assume that Algorithm* INB *does not break down. If x_* is a limit point of $\{x_k\}$ such that $F'(x_*)$ is invertible, then $F(x_*) = 0$ and $x_k \to x_*$. Furthermore, $s_k = \bar{s}_k$ and $\eta_k = \bar{\eta}_k$ for all sufficiently large k.*

Proof. Set $K \equiv \|F'(x_*)^{-1}\|$ and let $\delta > 0$ be sufficiently small that $F'(x)^{-1}$ exists and $\|F'(x)^{-1}\| \le 2K$ whenever $x \in N_\delta(x_*)$. Suppose that $x_k \in N_\delta(x_*)$, and set $r \equiv F(x_k) + F'(x_k)\,\bar{s}_k$. With $\sigma_k(\eta)$ given by (6.1), we have

$$\begin{aligned}
\|\sigma_k(\eta)\| &\le \|F'(x_k)^{-1}\|\,\|F'(x_k)\sigma_k(\eta)\| \\
&\le 2K\frac{1-\eta}{1-\bar{\eta}_k}\|-F(x_k)+r\| \\
&\le 2K\frac{1+\bar{\eta}_k}{1-\bar{\eta}_k}(1-\eta)\|F(x_k)\|
\end{aligned}$$

$$\leq 2K\frac{1+\eta_{\max}}{1-\eta_{\max}}(1-\eta)\|F(x_k)\|.$$

Thus (5.3) holds with $\Gamma \equiv 2K(1+\eta_{\max})/(1-\eta_{\max})$, and the theorem follows from Theorem 5.2. □

Application. Backtracking for (exact) Newton's method. When the initial trial step is the Newton step at each iteration, Algorithm INB has a particularly simple form. We take $\eta_{\max} = 0$, so that $\bar{\eta}_k = 0$ and $F'(x_k)\bar{s}_k = -F(x_k)$ for each k. Since $s_k = \sigma_k(\eta_k)$ for σ_k given by (6.1), we have $F'(x_k)s_k = -(1-\eta_k)F(x_k)$, whence

$$\|F(x_k) + F'(x_k)s_k\| = \eta_k\|F(x_k)\|.$$

Thus $pred_k(s_k) = (1-\eta_k)\|F(x_k)\|$, and the while-loop terminates when $ared_k(s_k) \geq t \cdot pred_k(s_k)$ (see §2). Algorithm INB can now be rephrased as follows:

ALGORITHM ENB (exact Newton method with backtracking).

LET x_0, $t \in (0,1)$, AND $0 < \theta_{\min} < \theta_{\max} < 1$ BE GIVEN.
FOR $k = 0$ STEP 1 UNTIL ∞ DO:
  SOLVE $F'(x_k)\, s_k = -F(x_k)$.
  WHILE $ared_k(s_k) < t \cdot pred_k(s_k)$ DO:
    CHOOSE $\theta \in [\theta_{\min}, \theta_{\max}]$.
    UPDATE $s_k \leftarrow \theta s_k$.
  SET $x_{k+1} = x_k + s_k$.

COROLLARY 6.2 (global convergence of algorithm ENB). *Assume that Algorithm* ENB *does not break down. If x_* is a limit point of $\{x_k\}$ such that $F'(x_*)$ is invertible, then $F(x_*) = 0$, $x_k \to x_*$, and $s_k = -F'(x_k)^{-1}F(x_k)$, the full Newton step, for all sufficiently large k.*

Extension. Piecewise linear backtracking through inexact Newton steps. Suppose that at the kth iteration of Algorithm MR we have choosen $\bar{\eta}_k$ and have determined inexact Newton steps $s_k^{(1)}, \ldots, s_k^{(m_k)}$ such that

$$\|F(x_k) + F'(x_k)\, s_k^{(j)}\| \leq \eta_k^{(j)}\|F(x_k)\|, \qquad j = 1, \ldots, m_k,$$

where $\eta_k^{(1)}, \ldots, \eta_k^{(m_k)}$ satisfy

$$\eta_{\max} \geq \eta_k^{(1)} > \cdots > \eta_k^{(m_k)} = \bar{\eta}_k.$$

We set $\eta_k^{(0)} \equiv 1$ and $s_k^{(0)} \equiv 0$ and define a piecewise linear curve σ_k by connecting the steps $s_k^{(j)}$:

$$\sigma_k(\eta) \equiv \frac{\eta - \eta_k^{(j)}}{\eta_k^{(j-1)} - \eta_k^{(j)}}\, s_k^{(j-1)} + \frac{\eta_k^{(j-1)} - \eta}{\eta_k^{(j-1)} - \eta_k^{(j)}}\, s_k^{(j)} \quad \text{if } \eta \in [\eta_k^{(j)}, \eta_k^{(j-1)}]. \tag{6.2}$$

We have (5.1) by convexity, and so this σ_k is admissible in Algorithm MR.

THEOREM 6.3. *Assume that Algorithm* MR *with each σ_k defined by* (6.2) *does not break down. If x_* is a limit point of $\{x_k\}$ such that $F'(x_*)$ is invertible, then $F(x_*) = 0$ and $x_k \to x_*$. Furthermore, $\eta_k = \bar{\eta}_k$ for all sufficiently large k.*

Proof. We claim that if $F'(x_k)$ is invertible, then

$$\|\sigma_k(\eta)\| \leq \|F'(x_k)^{-1}\| \left(\frac{1+\eta_{\max}}{1-\eta_{\max}}\right)(1-\eta)\|F(x_k)\|, \qquad \bar{\eta}_k \leq \eta \leq 1. \tag{6.3}$$

Indeed, if $\bar{\eta}_k \le \eta \le \eta_k^{(1)}$ $(\le \eta_{\max})$, then

$$\begin{aligned}\|F'(x_k)\,\sigma_k(\eta)\| &\le \|F(x_k)\| + \|F(x_k) + F'(x_k)\,\sigma_k(\eta)\| \\ &\le (1+\eta)\|F(x_k)\| \\ &\le \frac{1+\eta_{\max}}{1-\eta_{\max}}\,(1-\eta)\|F(x_k)\|;\end{aligned}$$

and if $\eta_k^{(1)} \le \eta \le \eta_k^{(0)}$ $(= 1)$, then

$$\begin{aligned}\|F'(x_k)\,\sigma_k(\eta)\| &= \left\| F'(x_k)\left[\frac{1-\eta}{1-\eta_k^{(1)}}\,s_k^{(1)}\right]\right\| \\ &\le \frac{1-\eta}{1-\eta_k^{(1)}}\left\{\|F(x_k)\| + \|F(x_k) + F'(x_k)\,s_k^{(1)}\|\right\} \\ &\le \frac{1-\eta}{1-\eta_k^{(1)}}\,(1+\eta_k^{(1)})\|F(x_k)\| \\ &\le \frac{1+\eta_{\max}}{1-\eta_{\max}}\,(1-\eta)\|F(x_k)\|.\end{aligned}$$

Since $\|\sigma_k(\eta)\| \le \|F'(x_k)^{-1}\|\,\|F'(x_k)\,\sigma_k(\eta)\|$, inequality (6.3) follows.

It is clear from (6.3) that an inequality of the form (5.3) holds for x_k sufficiently near a point x_* such that $F'(x_*)$ is invertible, and the theorem follows from Theorem 5.2. □

7. Equality curve methods. In this section, we consider instances of Algorithms MR and TL in which each curve σ_k is continuous, satisfies $\sigma_k(1) = 0$, and is such that (5.1) holds with equality, i.e.,

$$\|F(x_k) + F'(x_k)\,\sigma_k(\eta)\| = \eta\|F(x_k)\|, \qquad \bar{\eta}_k \le \eta \le 1. \tag{7.1}$$

In §8 we give important applications in which such σ_k's are easily determined. Here, we reformulate Algorithms MR and TL appropriately for such σ_k's and establish a somewhat specialized global convergence analysis, assuming that the norm is an inner product norm. These reformulated algorithms can be readily implemented as practical algorithms once a means of determining such σ_k's has been specified.

At the kth iteration of Algorithms MR and TL, the while-loop terminates if

$$\|F(x_k + \sigma_k(\eta_k))\| \le [1 - t(1-\eta_k)]\,\|F(x_k)\|. \tag{7.2}$$

By analogy with (2.4), we define

$$\begin{aligned}ared_k(\eta_k) &\equiv \|F(x_k)\| - \|F(x_k + \sigma_k(\eta_k))\|, \\ pred_k(\eta_k) &\equiv \|F(x_k)\| - \|F(x_k) + F'(x_k)\sigma_k(\eta_k)\|.\end{aligned}$$

If (7.1) holds, then $pred_k(\eta_k) = (1-\eta_k)\|F(x_k)\|$, and (7.2) is equivalent to

$$ared_k(\eta_k) \ge t \cdot pred_k(\eta_k).$$

This leads to the following reformulations of Algorithms MR and TL.

ALGORITHM ECMR (equality curve minimum reduction method).

LET x_0, $\eta_{\max} \in [0,1)$, $t \in (0,1)$, AND $0 < \theta_{\min} < \theta_{\max} < 1$ BE GIVEN.
FOR $k = 0$ STEP 1 UNTIL ∞ DO:
  CHOOSE $\bar{\eta}_k \in [0, \eta_{\max}]$.
  DETERMINE A CONTINUOUS σ_k SUCH THAT (7.1) HOLDS
    AND $\sigma_k(1) = 0$; SET $\eta_k = \bar{\eta}_k$.
  WHILE $ared_k(\eta_k) < t \cdot pred_k(\eta_k)$ DO:
    CHOOSE $\theta \in [\theta_{\min}, \theta_{\max}]$.
    UPDATE $\eta_k \leftarrow 1 - \theta(1 - \eta_k)$.
  SET $x_{k+1} = x_k + \sigma_k(\eta_k)$.

ALGORITHM ECTL (equality curve trust level method).

LET x_0, $\bar{\eta}_0 \in [0,1)$, $0 < t \le u < 1$, AND $0 < \theta_{\min} < \theta_{\max} < 1$ BE GIVEN.
FOR $k = 0$ STEP 1 UNTIL ∞ DO:
  DETERMINE A CONTINUOUS σ_k SUCH THAT (7.1) HOLDS
    AND $\sigma_k(1) = 0$; SET $\eta_k = \bar{\eta}_k$.
  WHILE $ared_k(\eta_k) < t \cdot pred_k(\eta_k)$ DO:
    CHOOSE $\theta \in [\theta_{\min}, \theta_{\max}]$.
    UPDATE $\eta_k \leftarrow 1 - \theta(1 - \eta_k)$.
  SET $x_{k+1} = x_k + \sigma_k(\eta_k)$.
  IF $ared_k(\eta_k) \ge u \cdot pred_k(\eta_k)$ CHOOSE $\bar{\eta}_{k+1} \in [0, \eta_k]$;
    ELSE CHOOSE $\bar{\eta}_{k+1} \in [0, 1 - \theta_{\min}(1 - \eta_k)]$.

Theorem 5.2 provides a general global convergence result for Algorithms ECMR and ECTL, but we develop a more specialized result under the assumption that the norm $\|\cdot\|$ is induced by an inner product $\langle \cdot, \cdot \rangle$, i.e., $\|v\| = \langle v, v \rangle^{1/2}$ for $v \in \mathbf{R}^n$.

Define $f(x) \equiv \frac{1}{2}\|F(x)\|^2$ for $x \in \mathbf{R}^n$ and note that the Fréchet derivative f' of f at x is given by $f'(x) = \langle F'(x)^* F(x), \cdot \rangle$, where $F'(x)^*$ is the adjoint operator of $F'(x)$ with respect to $\langle \cdot, \cdot \rangle$. Thus $F'(x)^* F(x)$ is the *gradient* of f at x with respect to $\langle \cdot, \cdot \rangle$, and $-F'(x)^* F(x)$ is the *steepest descent direction* (see Kantorovich and Akilov [11, pp. 462–463]).

LEMMA 7.1. *If x and s are such that $\|F(x) + F'(x)\, s\| < \|F(x)\|$, then*

$$\langle F'(x)^* F(x), s \rangle < 0,$$

i.e., s is a descent direction for f at x.

Proof. Writing $r \equiv F(x) + F'(x)\, s$, we have

$$\langle F'(x)^* F(x), s \rangle = \langle F(x), -F(x) + r \rangle \le -\|F(x)\|^2 + \|F(x)\|\|r\| < 0. \qquad \square$$

LEMMA 7.2. *At the kth step of Algorithm* ECMR (*Algorithm* ECTL), *assume that $F(x_k) \ne 0$ and set*

$$\gamma_k \equiv \inf_{\bar{\eta}_k \le \eta < 1} \frac{|\langle F(x_k), F'(x_k)\, \sigma_k(\eta) \rangle|}{\|F(x_k)\|\|F'(x_k)\, \sigma_k(\eta)\|}. \tag{7.3}$$

If $\gamma_k > 0$ and $F'(x_k)$ is invertible, then

$$\|\sigma_k(\eta)\| \le \frac{2\|F'(x_k)^{-1}\|}{\gamma_k}(1 - \eta)\|F(x_k)\| \tag{7.4}$$

for $\max\{\bar{\eta}_k, (1 - \gamma_k^2)^{1/2}\} \le \eta \le 1$.

Proof. For $\bar{\eta}_k \leq \eta \leq 1$, we have

$$\begin{aligned} \eta^2 \|F(x_k)\|^2 &= \|F(x_k) + F'(x_k)\,\sigma_k(\eta)\|^2 \\ &= \|F(x_k)\|^2 + 2\langle F(x_k), F'(x_k)\,\sigma_k(\eta)\rangle + \|F'(x_k)\,\sigma_k(\eta)\|^2. \end{aligned} \tag{7.5}$$

Define

$$\gamma(\eta) \equiv \frac{|\langle F(x_k), F'(x_k)\,\sigma_k(\eta)\rangle|}{\|F(x_k)\|\|F'(x_k)\,\sigma_k(\eta)\|}$$

and note that $\langle F(x_k), F'(x_k)\,\sigma_k(\eta)\rangle < 0$ by Lemma 7.1. Then (7.5) gives

$$\|F'(x_k)\,\sigma_k(\eta)\|^2 - 2\gamma(\eta)\|F(x_k)\|\|F'(x_k)\,\sigma_k(\eta)\| + (1-\eta^2)\|F(x_k)\|^2 = 0.$$

This is a quadratic equation in $\|F'(x_k)\,\sigma_k(\eta)\|$, the solutions of which are

$$\gamma(\eta)\|F(x_k)\| \left\{ 1 \pm \sqrt{1 - \frac{1-\eta^2}{\gamma(\eta)^2}} \right\}.$$

Since $\sigma_k(\eta)$ is continuous and $\sigma_k(1) = 0$, we must have

$$\|F'(x_k)\,\sigma_k(\eta)\| = \gamma(\eta)\|F(x_k)\| \left\{ 1 - \sqrt{1 - \frac{1-\eta^2}{\gamma(\eta)^2}} \right\} \tag{7.6}$$

whenever η is sufficiently near one. In particular, since $(1-\eta^2)/\gamma(\eta)^2 < 1$ for $\max\{\bar{\eta}_k, (1-\gamma_k^2)^{1/2}\} < \eta < 1$, we must have (7.6) for $\max\{\bar{\eta}_k, (1-\gamma_k^2)^{1/2}\} \leq \eta < 1$. Since $1 - \sqrt{1-\epsilon} \leq \epsilon$ whenever $0 \leq \epsilon \leq 1$, it follows that

$$\|F'(x_k)\,\sigma_k(\eta)\| \leq \gamma(\eta)\|F(x_k)\| \frac{1-\eta^2}{\gamma(\eta)^2} \leq \frac{2}{\gamma(\eta)}(1-\eta)\|F(x_k)\|$$

for $\max\{\bar{\eta}_k, (1-\gamma_k^2)^{1/2}\} \leq \eta < 1$. Since $\|\sigma_k(\eta)\| \leq \|F'(x_k)^{-1}\|\,\|F'(x_k)\,\sigma_k(\eta)\|$, inequality (7.4) follows. □

It is clear from (7.3) that $\gamma_k > 0$ if and only if the vectors $F'(x_k)\,\sigma_k(\eta)$ are bounded away from orthogonality with $F(x_k)$. Moreover, if $F'(x_k)$ is invertible, then $\gamma_k > 0$ if and only if the vectors $\sigma_k(\eta)$, which are descent directions for f at x_k, are bounded away from orthogonality with the steepest descent direction $-F'(x_k)^*\,F(x_k)$. This follows from

$$\begin{aligned} \kappa(F'(x_k))^{-1} \frac{|\langle F'(x_k)^*\,F(x_k), \sigma_k(\eta)\rangle|}{\|F'(x_k)^*\,F(x_k)\|\|\sigma_k(\eta)\|} &\leq \frac{|\langle F(x_k), F'(x_k)\,\sigma_k(\eta)\rangle|}{\|F(x_k)\|\|F'(x_k)\,\sigma_k(\eta)\|} \\ &\leq \kappa(F'(x_k)) \frac{|\langle F'(x_k)^*\,F(x_k), \sigma_k(\eta)\rangle|}{\|F'(x_k)^*\,F(x_k)\|\|\sigma_k(\eta)\|}, \end{aligned}$$

where $\kappa(F'(x_k)) \equiv \|F'(x_k)\|\|F'(x_k)^{-1}\|$ is the condition number of $F'(x_k)$.

If $F(x_k) \neq 0$, $F'(x_k)$ is invertible, and σ_k is such that $\gamma_k > 0$, then it follows from Lemmas 7.2 and 5.1 that neither Algorithm ECMR nor Algorithm ECTL breaks down in the while-loop. In fact, if $F'(x_k)$ is invertible, then it is always possible to avoid breakdown in either algorithm. Indeed, one can choose $\sigma_k(\eta) \equiv (1-\eta)[-F'(x_k)^{-1}F(x_k)]$, for which (7.1) holds. For this σ_k, if $F(x_k) \neq 0$, then

$\gamma_k = 1$, and if $F(x_k) = 0$, then $\sigma_k(\eta) \equiv 0$. In either case, neither algorithm breaks down in the while-loop.

The specialized global convergence result below is an immediate corollary of Theorem 5.2 and Lemma 7.2.

THEOREM 7.3 (global convergence of algorithms ECMR and ECTL). *Assume that Algorithm* ECMR (*Algorithm* ECTL) *does not break down. If x_* is a limit point of $\{x_k\}$ such that $F'(x_*)$ is invertible and if there exists a $\gamma > 0$ independent of k such that*

$$\gamma_k \equiv \inf_{\bar{\eta}_k \le \eta < 1} \frac{|\langle F(x_k), F'(x_k)\,\sigma_k(\eta)\rangle|}{\|F(x_k)\|\|F'(x_k)\,\sigma_k(\eta)\|} \ge \gamma \tag{7.7}$$

whenever $F(x_k) \neq 0$ and x_k is sufficiently near x_*, then $F(x_*) = 0$ and $x_k \to x_*$. Furthermore, $\eta_k = \bar{\eta}_k$ for all sufficiently large k.

Proof. Set $K \equiv \|F'(x_*)^{-1}\|$ and assume that $\delta > 0$ is sufficiently small that $F'(x)^{-1}$ exists and $\|F'(x)^{-1}\| \le 2K$ whenever $x \in N_\delta(x_*)$ and, furthermore, that (7.7) holds whenever $F(x_k) \neq 0$ and $x_k \in N_\delta(x_*)$.

Suppose that $x_k \in N_\delta(x_*)$. If $F(x_k) \neq 0$, then (7.7) and Lemma 7.2 give

$$\|\sigma_k(\eta)\| \le \frac{4K}{\gamma}(1-\eta)\|F(x_k)\| \tag{7.8}$$

for $\max\{\bar{\eta}_k, (1-\gamma^2)^{1/2}\} \le \eta \le 1$. If $F(x_k) = 0$, then $\sigma_k(\eta) \equiv 0$ since $F'(x_k)$ is invertible; thus equation (7.8) still holds. It follows easily that there is a Γ for which (5.3) holds whenever $x_k \in N_\delta(x_*)$, and the theorem follows from Theorem 5.2. □

8. Applications of equality curve methods. In this section, we give applications of the results in §7, focusing on ways of determining the curves σ_k in Algorithms ECMR and ECTL. In the first application, our main purpose is to outline ways of determining these curves in certain Newton iterative methods; we also show how to obtain a dogleg globalization of (exact) Newton's method using Algorithm ECTL. In the second application, we show how Levenberg–Marquardt globalizations of Newton's method can be obtained from Algorithms ECMR and ECTL.

We assume throughout that $\|\cdot\|$ is induced by an inner product $\langle\cdot,\cdot\rangle$.

Application 1. Piecewise linear backtracking through residual minimizing steps. Suppose that the Newton equation (1.2) is solved approximately using a *residual minimizing* iterative method, i.e., a method for solving a linear system $Au = b$ that begins with an initial approximate solution u_0 (and residual $r_0 \equiv b - Au_0$) and, at the ith iteration, determines a subspace $\mathcal{K}_i \subseteq \mathbf{R}^n$ and a correction

$$z_i \in \arg\min_{z \in \mathcal{K}_i} \|b - A(u_0 + z)\|.$$

There are many such methods. Two familiar examples, in which $\|\cdot\| = \|\cdot\|_2$, are the GMRES method of Saad and Schultz [15], with $\mathcal{K}_i \equiv \operatorname{span}\{r_0, Ar_0, \ldots, A^{i-1}r_0\}$, and the conjugate gradient method applied to the normal equations $A^TAx = A^Tb$, or CGNR (see Elman [9]), with $\mathcal{K}_i \equiv \operatorname{span}\{A^Tr_0, (A^TA)A^Tr_0, \ldots, (A^TA)^{i-1}A^Tr_0\}$.

Here, we require only that $\mathcal{K}_1 \subset \mathcal{K}_2 \subset \cdots$. Then $\rho_i \equiv \|b - A(u_0 + z_i)\|$ is nonincreasing in i. The following lemma shows that if $\rho_i > \rho_j$ for some $i < j$, then the residual norm is strictly monotone decreasing along the line segment from $u_0 + z_i$ to $u_0 + z_j$ and gives a parametrization of this line segment in terms of the residual norm.

LEMMA 8.1. *Suppose that* $\rho_i > \rho_j$. *For* $\rho_i \geq \rho \geq \rho_j$, *set*

$$\tau(\rho) \equiv \left(\frac{\rho^2 - \rho_j^2}{\rho_i^2 - \rho_j^2} \right)^{\frac{1}{2}} \quad \text{and} \quad z(\rho) \equiv \tau(\rho) z_i + (1 - \tau(\rho)) z_j.$$

Then $\|b - A(u_0 + z(\rho))\| = \rho$ *for* $\rho_i \geq \rho \geq \rho_j$.

Proof. Defining $\|w\|_A \equiv \|Aw\|$ for $w \in \mathbf{R}^n$, we have

$$\|b - A(u_0 + z)\| = \|u - u_0 - z\|_A.$$

Therefore, for any l, we have $z_l = P_l(u - u_0)$, where P_l denotes $\|\cdot\|_A$-orthogonal projection onto $\mathcal{K}_l$, and

$$\rho_l = \|u - u_0 - z_l\|_A = \|P_l^{\perp}(u - u_0)\|_A,$$

where $P_l^{\perp} \equiv I - P_l$. Also, since $i < j$, we have $\mathcal{K}_i \subset \mathcal{K}_j$ and $P_j^{\perp} P_i^{\perp} = P_i^{\perp} P_j^{\perp} = P_j^{\perp}$. It follows that for $\rho_i \geq \rho \geq \rho_j$,

$$\begin{aligned}
\|b - A(u_0 + z(\rho))\|^2 &= \|u - u_0 - z(\rho)\|_A^2 \\
&= \|\tau(\rho) P_i^{\perp}(u - u_0) + (1 - \tau(\rho)) P_j^{\perp}(u - u_0)\|_A^2 \\
&= \tau(\rho)^2 \|P_i^{\perp}(u - u_0)\|_A^2 + (1 - \tau(\rho)^2) \|P_j^{\perp}(u - u_0)\|_A^2 \\
&= \tau(\rho)^2 \rho_i^2 + (1 - \tau(\rho)^2) \rho_j^2 \\
&= \rho^2. \qquad \square
\end{aligned}$$

We determine a curve σ_k at the kth step of Algorithm ECMR or Algorithm ECTL as follows: If $F(x_k) = 0$, then $\sigma_k(\eta) \equiv 0$ for $\bar{\eta}_k \leq \eta \leq 1$. If $F(x_k) \neq 0$, then we apply a residual minimizing method to the Newton equation (1.2), beginning with zero as the initial approximate solution, to obtain approximate solutions $s_k^{(0)} = 0$, $s_k^{(1)}, \ldots, s_k^{(m_k)}$, which are not necessarily *consecutive* iterates but such that

$$1 = \eta_k^{(0)} > \eta_k^{(1)} > \cdots > \eta_k^{(m_k - 1)} > \bar{\eta}_k \geq \eta_k^{(m_k)},$$

where

$$\eta_k^{(i)} \equiv \frac{\|F(x_k) + F'(x_k)\, s_k^{(i)}\|}{\|F(x_k)\|}, \qquad i = 0, \ldots, m_k.$$

If $\bar{\eta}_k > \eta_k^{(m_k)}$, then we force $\bar{\eta}_k = \eta_k^{(m_k)}$ either by redefining $\bar{\eta}_k$ to be $\eta_k^{(m_k)}$ or by replacing $s_k^{(m_k)}$ with an appropriate convex combination of $s_k^{(m_k)}$ and $s_k^{(m_k - 1)}$, which is easily done using Lemma 8.1. We take σ_k to be the piecewise linear curve connecting $s_k^{(0)}, \ldots, s_k^{(m_k)}$. It follows from Lemma 8.1 that this curve can easily be parametrized continuously in η such that (7.1) holds and $\sigma_k(1) = 0$.

We give global convergence results for Algorithms ECMR and ECTL with each σ_k determined in this way.

LEMMA 8.2. *If* $\|F(x_k)\| \neq 0$ *and* $\|F(x_k) + F'(x_k)\, s\| \leq \eta \|F(x_k)\|$ *for some* s *and* $\eta \in [0, 1)$, *then*

$$\frac{|\langle F(x_k), F'(x_k)\, s \rangle|}{\|F(x_k)\| \|F'(x_k)\, s\|} \geq \frac{1 - \eta}{1 + \eta}.$$

Proof. Writing $r \equiv F(x_k) + F'(x_k)\, s$, we have $\|r\| \le \eta \|F(x_k)\| < \|F(x_k)\|$. Then $F'(x_k)\, s \neq 0$, and

$$\frac{|\langle F(x_k), F'(x_k)\, s\rangle|}{\|F(x_k)\|\|F'(x_k)\, s\|} = \frac{|\langle F(x_k), -F(x_k) + r\rangle|}{\|F(x_k)\|\|-F(x_k) + r\|}$$

$$\ge \frac{\|F(x_k)\|^2 - \|F(x_k)\|\|r\|}{\|F(x_k)\|\,(\|F(x_k)\| + \|r\|)} \ge \frac{1-\eta}{1+\eta}. \qquad \square$$

THEOREM 8.3. *Assume that Algorithm* ECMR *with each σ_k determined as above does not break down and that $\eta_k^{(1)} \le \eta_{\max}$ whenever $F(x_k) \neq 0$. If x_* is a limit point of $\{x_k\}$ such that $F'(x_*)$ is invertible, then $F(x_*) = 0$ and $x_k \to x_*$. Furthermore, $\eta_k = \bar{\eta}_k$ for all sufficiently large k.*

Proof. We claim that if $F(x_k) \neq 0$ and $\eta_k^{(1)} \le \eta_{\max}$, then

$$\gamma_k \equiv \inf_{\bar{\eta}_k \le \eta < 1} \frac{|\langle F(x_k), F'(x_k)\, \sigma_k(\eta)\rangle|}{\|F(x_k)\|\|F'(x_k)\, \sigma_k(\eta)\|} \ge \gamma \equiv \frac{1-\eta_{\max}}{1+\eta_{\max}}.$$

Indeed, if $\eta_k^{(1)} \le \eta < 1$, then $\sigma_k(\eta) = \tau s_k^{(1)}$ for some $\tau \in (0, 1]$, and Lemma 8.2 gives

$$\frac{|\langle F(x_k), F'(x_k)\, \sigma_k(\eta)\rangle|}{\|F(x_k)\|\|F'(x_k)\, \sigma_k(\eta)\|} = \frac{|\langle F(x_k), F'(x_k)\, s_k^{(1)}\rangle|}{\|F(x_k)\|\|F'(x_k)\, s_k^{(1)}\|} \ge \frac{1-\eta_k^{(1)}}{1+\eta_k^{(1)}} \ge \frac{1-\eta_{\max}}{1+\eta_{\max}}.$$

If $\bar{\eta}_k \le \eta \le \eta_k^{(1)}$, then it follows from Lemma 8.2 that

$$\frac{|\langle F(x_k), F'(x_k)\, \sigma_k(\eta)\rangle|}{\|F(x_k)\|\|F'(x_k)\, \sigma_k(\eta)\|} \ge \frac{1-\eta}{1+\eta} \ge \frac{1-\eta_k^{(1)}}{1+\eta_k^{(1)}} \ge \frac{1-\eta_{\max}}{1+\eta_{\max}}.$$

The theorem now follows from Theorem 7.3. $\square$

To treat Algorithm ECTL, we formulate the following condition on F and the residual minimizing method (with zero as the initial approximate solution) near a point x_*.

Nonorthogonality condition. There exist $\delta > 0$ and $\lambda > 0$ such that if $x \in N_\delta(x_*)$ and $F(x) \neq 0$, then $F'(x)\, z \neq 0$ for *some* $z \in \mathcal{K}_1$ and

$$\max_{\substack{z \in \mathcal{K}_1 \\ F'(x)\, z = 0}} \frac{|\langle F(x), F'(x)\, z\rangle|}{\|F(x)\|\|F'(x)\, z\|} \ge \lambda.$$

Remark. This is a condition that $F'(x)(\mathcal{K}_1)$ be uniformly bounded away from orthogonality with $F(x)$ for x near x_*. As in the remark after Lemma 7.2, if $F'(x)$ is invertible near x_*, then it is equivalent to a condition that $\mathcal{K}_1$ be uniformly bounded away from orthogonality with $-F'(x)^*\, F(x)$, the steepest descent direction for $f(x) \equiv \frac{1}{2}\|F(x)\|^2$, for x near x_*. Since F' is continuous, this condition holds, e.g., if the method is CGNR and $F'(x_*)$ is invertible, or if the method is GMRES and $F'(x_*) + F'(x_*)^T$ is positive definite.

THEOREM 8.4. *Assume that Algorithm* ECTL *with each σ_k determined as above does not break down. If x_* is a limit point of $\{x_k\}$ such that $F'(x_*)$ is invertible and the nonorthogonality condition holds, then $F(x_*) = 0$ and $x_k \to x_*$. Furthermore, $\eta_k = \bar{\eta}_k$ for all sufficiently large k.*

Proof. Let $\delta > 0$ and $\lambda \in [0,1)$ be as in the nonorthogonality condition, and assume that $x_k \in N_\delta(x_*)$ and $F(x_k) \neq 0$. By the nonorthogonality condition, we have

$$\begin{aligned}
\min_{z\in\mathcal{K}_1} \|F(x_k) + F'(x_k)\,z\| &= \min_{\substack{z\in\mathcal{K}_1 \\ F'(x_k)z=0}} \|F(x_k) + F'(x_k)\,z\| \\
&= \min_{\substack{z\in\mathcal{K}_1 \\ F'(x_k)z=0}} \min_{\mu\in\mathbf{R}} \|F(x_k) + F'(x_k)\,(\mu z)\| \\
&= \min_{\substack{z\in\mathcal{K}_1 \\ F'(x_k)z=0}} \left(\|F(x_k)\|^2 - \frac{\langle F(x_k), F'(x_k)\,z\rangle^2}{\|F'(x_k)\,z\|^2}\right)^{\frac{1}{2}} \\
&= \left(1 - \max_{\substack{z\in\mathcal{K}_1 \\ F'(x_k)z=0}} \frac{\langle F(x_k), F'(x_k)\,z\rangle^2}{\|F(x_k)\|^2\|F'(x_k)\,z\|^2}\right)^{\frac{1}{2}} \|F(x_k)\| \\
&\leq \sqrt{1-\lambda^2}\,\|F(x_k)\|.
\end{aligned}$$

Then we must have $\eta_k^{(1)} \leq \sqrt{1-\lambda^2}$ and, by an argument similar to that used in the proof of Theorem 8.3,

$$\gamma_k \equiv \inf_{\bar\eta_k\leq\eta<1} \frac{|\langle F(x_k), F'(x_k)\,\sigma_k(\eta)\rangle|}{\|F(x_k)\|\|F'(x_k)\,\sigma_k(\eta)\|} \geq \gamma \equiv \frac{1-\sqrt{1-\lambda^2}}{1+\sqrt{1-\lambda^2}}.$$

The theorem now follows from Theorem 7.3. □

The *dogleg curve* is widely used in trust region globalizations of (exact) Newton's method; see, e.g., Dennis and Schnabel [5, Chap. 6]. At x_k, this curve is the piecewise linear curve connecting zero, the minimizer of $\|F(x_k) + F'(x_k)\,s\|_2$ in the steepest descent direction $-F'(x_k)^T F(x_k)$ (the steepest descent step), and the Newton step $-F'(x_k)^{-1}F(x_k)$. With $\|\cdot\| = \|\cdot\|_2$, we can determine σ_k as a subset of this curve in Algorithm ECTL. Indeed, if we choose $\mathcal{K}_1 \equiv \text{span}\{-F'(x_k)^T F(x_k)\}$ and $\mathcal{K}_2 \equiv \mathbf{R}^n$ and take zero as the initial approximate solution, then the first and second residual minimizing iterates are the steepest descent step and the Newton step, respectively, and we obtain the dogleg curve.[2] Consequently, we can take σ_k to be the part of the dogleg curve from zero to the point at which the local linear model norm is $\bar\eta_k\|F(x_k)\|_2$. Since the nonorthogonality condition holds for this choice of $\mathcal{K}_1$ if $F'(x_*)$ is invertible, the corollary below follows immediately from Theorem 8.4.

COROLLARY 8.5. *Assume that Algorithm* ECTL *with each σ_k obtained from the dogleg curve does not break down. If x_* is a limit point of $\{x_k\}$ such that $F'(x_*)$ is invertible, then $F(x_*) = 0$ and $x_k \to x_*$. Furthermore, $\eta_k = \bar\eta_k$ for all sufficiently large k.*

Application 2. Levenberg–Marquardt globalizations of (exact) Newton's method. The *Levenberg–Marquardt curve* is also often used in trust region globalizations of Newton's method; see, e.g, Dennis and Schnabel [5, Chap. 6]. At x_k, this curve is swept out by $\arg\min_{\|s\|_2\leq\delta} \|F(x_k) + F'(x_k)\,s\|_2$, $0 < \delta < \infty$, and is usually given by

$$\sigma_k(\mu) \equiv -\left[F'(x_k)^T F'(x_k) + \mu I\right]^{-1} F'(x_k)^T F(x_k), \qquad 0 \leq \mu < \infty. \tag{8.1}$$

[2] The dogleg curve can also be obtained using CGNR: With zero as the initial approximate solution, the first CGNR iteration gives the steepest descent step, and the Newton step is obtained after no more than n iterations.

Note that $\sigma_k(0) = -F'(x_k)^{-1}F(x_k)$, the Newton step, and that $\lim_{\mu\to\infty}\sigma_k(\mu) = 0$. It is well known that

$$\eta(\mu) \equiv \frac{\|F(x_k) + F'(x_k)\,\sigma_k(\mu)\|_2}{\|F(x_k)\|_2}$$

is strictly monotone increasing in μ; furthermore, $\eta(0) = 0$ and $\lim_{\mu\to\infty}\eta(\mu) = 1$. Therefore, we can reparametrize the curve in η and, setting $\sigma_k(1) \equiv 0$, obtain a continuous curve σ_k such that $\sigma_k(1) = 0$ and

$$\|F(x_k) + F'(x_k)\,\sigma_k(\eta)\|_2 = \eta\|F(x_k)\|_2, \qquad 0 \le \eta \le 1.$$

We obtain Levenberg–Marquardt globalizations of Newton's method from Algorithms ECMR and ECTL by taking $\|\cdot\| = \|\cdot\|_2$ and taking each σ_k to be the subcurve of the Levenberg–Marquardt curve from zero to the point at which the local linear model norm is $\bar{\eta}_k\|F(x_k)\|_2$.

THEOREM 8.6. *Assume that Algorithm* ECMR (*Algorithm* ECTL) *with each σ_k obtained from the Levenberg–Marquardt curve does not break down. If x_* is a limit point of $\{x_k\}$ such that $F'(x_*)$ is invertible, then $F(x_*) = 0$ and $x_k \to x_*$. Furthermore, $\eta_k = \bar{\eta}_k$ for all sufficiently large k.*

Proof. Set $K \equiv \max\{\|F'(x_*)\|_2, \|F'(x_*)^{-1}\|_2\}$ and let $\delta > 0$ be sufficiently small that $F'(x)^{-1}$ exists and $\max\{\|F'(x)\|_2, \|F'(x)^{-1}\|_2\} \le 2K$ whenever $x \in N_\delta(x_*)$.

Suppose that $x_k \in N_\delta(x_*)$ and $F(x_k) \neq 0$. We derive a lower bound on γ_k in (7.7) with $\bar{\eta}_k = 0$; this will also be a lower bound for any $\bar{\eta}_k \in [0, 1)$. We work with the usual parametrization (8.1). Note that

$$\begin{aligned} F'(x_k)\,\sigma_k(\mu) &= -F'(x_k)\left[F'(x_k)^T F'(x_k) + \mu I\right]^{-1} F'(x_k)^T F(x_k) \\ &= -\left\{I + \mu\left[F'(x_k)F'(x_k)^T\right]^{-1}\right\}^{-1} F(x_k) \\ &= -Q\left(I + \mu\Lambda^{-1}\right)^{-1} Q^T F(x_k), \end{aligned} \tag{8.2}$$

where $F'(x_k)F'(x_k)^T = Q\Lambda Q^T$, Q is orthogonal, and $\Lambda = \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$ with each $\lambda_i > 0$. Setting $\lambda_{\max} \equiv \max_i \lambda_i$, $\lambda_{\min} \equiv \min_i \lambda_i$, and $w \equiv (w_1, \ldots, w_n)^T \equiv Q^T F(x_k)$, we have from (7.7) and (8.2) that

$$\begin{aligned} \gamma_k &= \inf_{0\le\mu<\infty} \frac{\left|w^T\left(I + \mu\Lambda^{-1}\right)^{-1} w\right|}{\|w\|_2\|\left(I + \mu\Lambda^{-1}\right)^{-1} w\|_2} \\ &= \inf_{0\le\mu<\infty} \frac{\sum_{i=1}^n \frac{\lambda_i}{\lambda_i+\mu} w_i^2}{\left(\sum_{i=1}^n w_i^2\right)^{1/2}\left[\sum_{i=1}^n \left(\frac{\lambda_i}{\lambda_i+\mu}\right)^2 w_i^2\right]^{1/2}} \\ &\ge \inf_{0\le\mu<\infty} \left(\frac{\lambda_{\min}}{\lambda_{\max}+\mu}\right) \Big/ \left(\frac{\lambda_{\max}}{\lambda_{\min}+\mu}\right) \\ &= \inf_{0\le\mu<\infty} \frac{\lambda_{\min}}{\lambda_{\max}} \frac{\lambda_{\min}+\mu}{\lambda_{\max}+\mu} \\ &\ge \left(\frac{\lambda_{\min}}{\lambda_{\max}}\right)^2 \\ &= \left(\frac{1}{\|F'(x_k)^{-1}\|_2\|F'(x_k)\|_2}\right)^4 \end{aligned}$$

$$\geq \frac{1}{256K^8}.$$

Thus (7.7) holds with $\gamma = (256K^8)^{-1}$ whenever $x_k \in N_\delta(x_*)$ and $F(x_k) \neq 0$, and the theorem follows from Theorem 7.3. □

9. Postview. In this paper, we have outlined inexact Newton methods with strong global convergence properties. Our purposes have been the following:

- to introduce new algorithms that will be useful in implementing Newton iterative methods and other Newton-like methods in which the Newton equation (1.2) is solved inaccurately;
- to provide a general framework for treating globalizations of Newton's method and related methods.

Our most general method is Algorithm GIN in §2; all subsequent algorithms are special cases of it. The analysis for Algorithm GIN in §3 provides the basis for all global convergence results in the sequel; for all methods, except the paradigm methods in §5, these include a basic result to the effect that, under reasonable assumptions, if a sequence of iterates has a limit point at which F' is invertible, then that limit point is a solution of (1.1) and the iterates converge to it. The developments in §§2–3 are first applied directly in §4 to give global convergence results for trust region methods and global approximate Newton methods. They are then used in §5 to develop a theoretical basis for Algorithms MR and TL, which, in turn, serve as paradigms for algorithms in §§6–8 that are of ultimate interest.

The algorithms in §§6–8 test initial inexact Newton steps and, if they are unsatisfactory, determine new steps in specified ways until satisfactory steps are found. For each of these algorithms, we show, in addition to the basic global convergence result above, that initial inexact Newton steps are ultimately satisfactory near a solution at which F' is invertible and, therefore, the convergence can be made as fast as desired, up to the rate of Newton's method, by choosing appropriately small initial inexact Newton levels.

The algorithms in §6 are backtracking methods. The principal method is Algorithm INB for backtracking from initial inexact Newton steps that achieve a specified minimum reduction of the local linear model norm but otherwise are arbitrary. This algorithm is based on Algorithm MR. It can easily be implemented as a practical algorithm once a means of determining initial inexact Newton steps is given. As a special case of Algorithm INB, we give a backtracking globalization of (exact) Newton's method in Algorithm ENB. We also discuss extensions of Algorithm INB to piecewise linear backtracking through inexact Newton steps.

The algorithms in §§7–8 are equality curve methods. The basic algorithms are Algorithms ECMR and ECTL in §7, which are based on Algorithms MR and TL, respectively. To implement these, one must provide a means of specifying continuous curves of inexact Newton steps along which inexact Newton conditions are satisfied with equality. Applications are given in §8 in which such curves are easily and naturally determined. The principal application is to Newton iterative methods in which the Newton equation (1.2) is solved approximately using a residual minimizing iterative method; we also show how to obtain dogleg and Levenberg–Marquardt globalizations of (exact) Newton's method.

We close with Fig. 1 below, which gives an organizational chart of the methods and applications in this paper and their interrelationships.

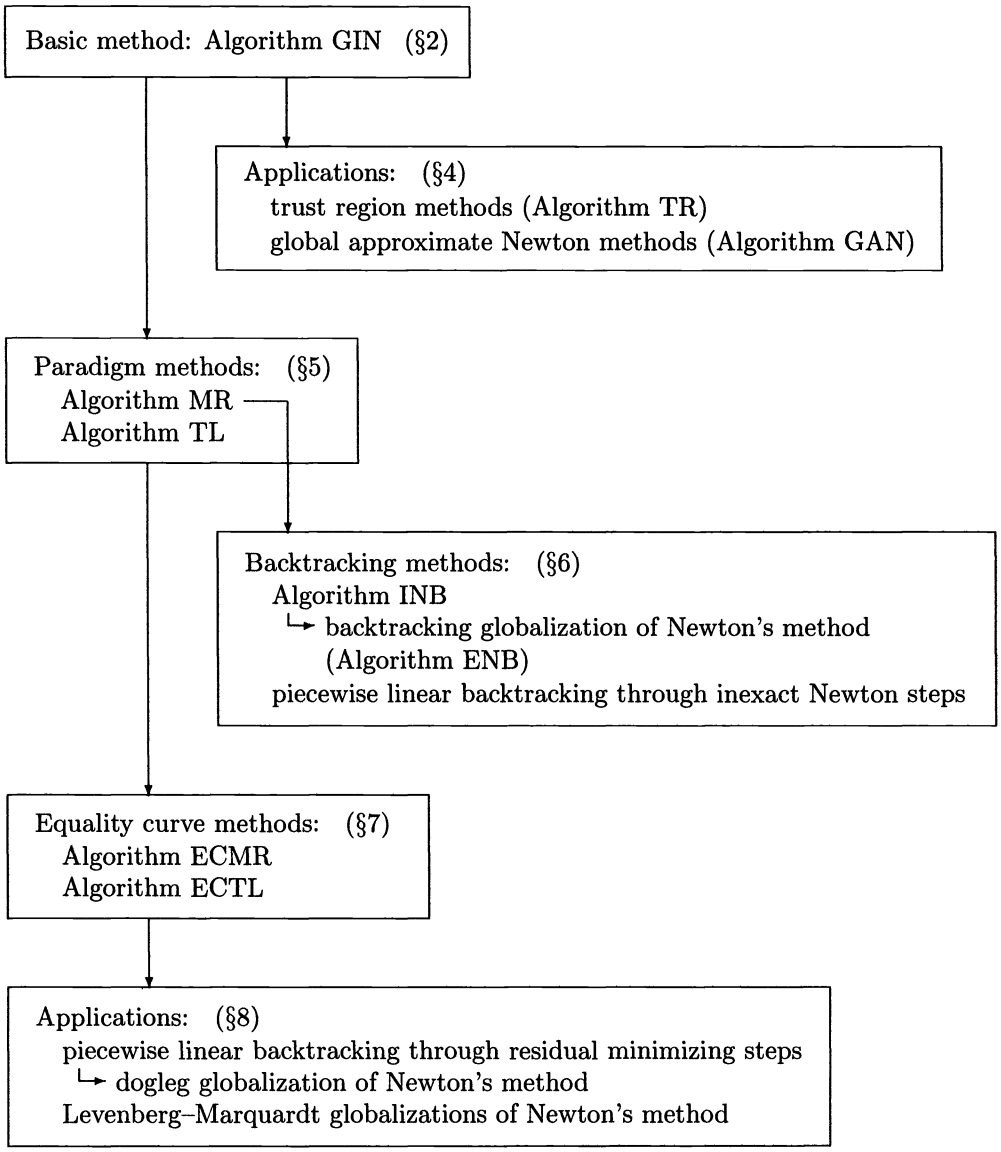


FIG. 1. *Organizational chart of algorithms and applications.*

REFERENCES

[1] L. ARMIJO, *Minimization of functions having Lipschitz-continuous first derivatives*, Pacific J. Math., 16 (1966), pp. 1–3.

[2] R. E. BANK AND D. J. ROSE, *Global approximate Newton methods*, Numer. Math., 37 (1981), pp. 279–295.

[3] P. N. BROWN AND Y. SAAD, *Convergence theory of nonlinear Newton–Krylov algorithms*, SIAM J. Optimization, 4 (1994), pp. 297–330.

[4] R. S. DEMBO, S. C. EISENSTAT, AND T. STEIHAUG, *Inexact Newton methods*, SIAM J. Numer. Anal., 19 (1982), pp. 400–408.

[5] J. E. DENNIS, JR. AND R. B. SCHNABEL, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Series in Automatic Computation, Prentice-Hall, Englewood

Cliffs, NJ, 1983.

[6] P. DEUFLHARD, *Global inexact Newton methods for very large scale nonlinear problems*, Impact Comput. Sci. Engrg., 3 (1991), pp. 366–393.

[7] M. EL HALLABI, *A Global Convergence Theory for Arbitrary Norm Trust Region Methods for Nonlinear Equations*, Ph.D. thesis, Department of Mathematical Sciences, Rice University, Houston, TX, 1987.

[8] M. EL HALLABI AND R. A. TAPIA, *A global convergence theory for arbitrary norm trust-region methods for nonlinear equations*, Tech. Report TR87-25, Department of Mathematical Sciences, Rice University, Houston, TX, November 1987; revised July, 1989.

[9] H. C. ELMAN, *Iterative Methods for Large, Sparse, Nonsymmetric Systems of Linear Equations*, Ph.D. thesis, Department of Computer Science, Yale University, New Haven, CT, 1982.

[10] A. A. GOLDSTEIN, *Constructive Real Analysis*, Harper and Row, New York, 1967.

[11] L. KANTOROVICH AND G. AKILOV, *Functional Analysis in Normed Spaces*, second ed., Pergamon Press Ltd., Oxford, 1982; Translated by H. L. Silcock, published in Russian by Nauka Publishers, Moscow, 1982.

[12] J. J. MORÉ AND D. C. SORENSEN, *Computing a trust region step*, SIAM J. Sci. Statist. Comput., 4 (1983), pp. 553–572.

[13] J. M. ORTEGA AND W. C. RHEINBOLDT, *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, New York, 1970.

[14] M. J. D. POWELL, *General algorithms for discrete nonlinear approximation calculations*, in Approximation Theory IV, C. K. Chui, L. L. Schumaker, and J. D. Ward, eds., 1983, Academic Press, New York, pp. 187–218.

[15] Y. SAAD AND M. H. SCHULTZ, *GMRES: A generalized minimal residual method for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869.

[16] G. A. SHULTZ, R. B. SCHNABEL, AND R. H. BYRD, *A family of trust-region-based algorithms for unconstrained minimization with strong global convergence properties*, SIAM J. Numer. Anal., 22 (1985), pp. 47–67.

[17] T. STEIHAUG, *Quasi-Newton Methods for Large Scale Nonlinear Problems*, Ph.D. thesis, School of Organization and Management, Yale University, New Haven, CT, 1981.

[18] P. WOLFE, *Convergence conditions for ascent methods*, SIAM Rev., 11 (1969), pp. 226–235.

[19] ———, *Convergence conditions for ascent methods. II: Some corrections*, SIAM Rev., 13 (1971), pp. 185–188.

SIAM J. OPTIMIZATION
Vol. 4, No. 2, pp. 423–440, May 1994
© 1994 Society for Industrial and Applied Mathematics
011

AN INTERIOR POINT COLUMN GENERATION METHOD FOR LINEAR PROGRAMMING USING SHIFTED BARRIERS*

JOHN E. MITCHELL†

Abstract. A column generation method for linear programming problems is described. This method is based on Freund's shifted barrier method for linear programming. The use of shifts makes it possible to give new primal variables the value zero, with the resulting point being both feasible and interior in the shifted or relaxed formulation. The algorithm takes $O(nL)$ iterations. It is shown that the direction obtained when using the shifted barrier column generation method is related to one proposed by Mitchell and Todd. Preliminary computational results are included, which show that the method may be useful when small numbers of variables are added.

Key words. linear programming, column generation, interior point algorithms, shifted barriers, polynomiality

AMS subject classifications. 90C05, 65K05, 49M27

1. Introduction.

We describe a column generation algorithm that is based on Freund's two "warm start" shifted barrier function algorithms for linear programming [4]. By using shifted barriers, extra variables can be introduced with value zero, and the new point will be an interior point.

Column generation methods have the potential to reduce considerably the computational time required to solve a linear programming problem with many more columns than rows. They can also result in a reduction in memory requirement, because columns need only be stored in main memory when they are included in the current relaxation. Generating columns corresponds to adding constraints in the dual linear program. A cutting plane algorithm for solving integer programming problems is one where constraints are added to a linear program as the solution proceeds. In such a method, a linear programming relaxation of the integer program is solved; if the solution is feasible in the integer program, we are done; otherwise, we generate a "cutting plane" or extra constraint that is added to the problem, and then this new modified linear programming problem is solved. The cutting plane is generated using a separation routine. Such methods have been used very successfully recently; see, for example, Grötschel and Holland [9] and Padberg and Rinaldi [14]. Other papers that have described using interior point methods in a column generation setting include those by Goffin, Haurie, and Vial [6], [5], Mitchell and Todd [13], Ye and Kaliski [18], [12], and Atkinson and Vaidya [1].

Freund's algorithms are based on his potential function reduction algorithms given in [3], which are similar to Ye's algorithm given in [17]. These algorithms use primal steps to decrease a potential function while good reductions can be obtained, and they use a dual step whenever the primal step cannot be guaranteed to give the necessary reduction. Freund gives two algorithms in [3], with one using a primal potential function and the other a primal-dual potential function. In [4] the nonnegativity constraints $x \geq 0$ are replaced by constraints of the form $x + h(c^T x - B) \geq 0$, where $c^T x - B$ is the current duality gap and h is a nonnegative vector. Thus, if the vector h is chosen appropriately, a certain amount of negativity is allowed in the variables x. One result of this is that it is possible to have "interior" points where the variables

* Received by the editors December 10, 1990; accepted for publication (in revised form) March 8, 1993. This research was partially supported by Office of Naval Research grant N00014–90–J–1714.
† Department of Mathematical Sciences, Rensselaer Polytechnic Institute, Troy, New York 12180.

are zero. This can be used in a column generation method: additional variables can be given the value zero when they enter.

In §2 we describe our column generation algorithm. Properties of the algorithm are discussed in §§3 and 4. The principal result is to relate the direction used when introducing a new variable to a direction described in [13]. Section 5 contains preliminary computational results.

Throughout, e denotes a vector of ones of the appropriate dimension. Given a lower-case vector such as $x = (x_1, \ldots, x_n)^T$, the diagonal matrix $\mathrm{diag}(x_1, \ldots, x_n)$ is denoted by the upper-case letter X. The matrix operator that projects a vector v onto the null space of a matrix M is denoted by P^M. Thus $P^M v = (I - M^T(MM^T)^{-1}M)v$ if M has full row rank.

2. The algorithm.

2.1. Freund's linear programming algorithm. Consider the standard form linear programming problem

$$\text{(P)} \qquad \begin{array}{ll} \min & c^T x \\ \text{s.t.} & Ax = b, \\ & x \geq 0. \end{array}$$

Here, A is an $m \times n$ matrix, c and x are n-vectors, and b is an m-vector. Following Freund [4], we consider the following relaxation of (P):

$$\text{(P(B))} \qquad \begin{array}{lll} \min & c^T x & \\ \text{s.t.} & Ax & = b, \\ & x + h(c^T x - B) & \geq 0. \end{array}$$

Here, B is a lower bound on the optimal value of (P) and h is a nonnegative n-vector satisfying $c^T h > -1$. Thus, h is a shift vector with shift parameter $c^T x - B$. Freund introduces a change of variables $y = T(x) := x + h(c^T x - B)$, and then (P(B)) is equivalent to the standard form problem

$$\text{(PM(B))} \qquad \begin{array}{ll} \min & c^T y/(1 + c^T h) \\ \text{s.t.} & AM^{-1}y = \bar{b}, \\ & y \geq 0, \end{array}$$

where $\bar{b} = b - (B/(1 + c^T h))Ah$ and $M = [I + hc^T]$, so $M^{-1} = [I - (1/(1 + c^T h))hc^T]$. It should be noted that $M^{-T}c = (1/(1+c^T h))c$, so the change of variables rescales the objective function, not otherwise altering it, and $c^T y = (1 + c^T h)c^T T^{-1}(y) + Bc^T h$. Similarly, $AM^{-1}h = (1/(1+c^T h))Ah$. The solution to (PM(B)) can be updated using a standard method, and Freund proposes using one of the two methods in [3]. For the sake of brevity, we restrict attention to the first of these methods. The second method uses a primal-dual potential function, so it is necessary to have a dual solution that is strictly feasible, that is, the inequality constraints are satisfied strictly. In certain settings, the possibility of selecting a dual feasible point on the boundary can be useful. For example, if the dual problem is the relaxation of an integer programming problem, it is often easy to get a dual feasible integral solution that is not strictly feasible.

Consider the potential function

$$F(x, B) := q \ln(c^T x - B) - \sum_{j=1}^{n} \ln(x_j + h_j(c^T x - B)),$$

where $q > n$ is a given parameter. The algorithm is an affine scaling descent algorithm applied to this potential function. A lower bound $\hat{B}$ on the optimal value of (P) is assumed known, and an initial primal solution $\hat{x}$ is given. In addition, it can be assumed that $\hat{x} + h(c^T\hat{x} - \hat{B}) > 0$, decreasing $\hat{B}$ if necessary. The parameter ϵ is a tolerance on optimality and γ is used to determine when to use a primal step and when to use a dual one. We require that $\gamma < 1$.

ALGORITHM 2.1 (FREUND).

1. Input data $(A, b, c, h, \hat{x}, \hat{B}, \epsilon, q, \gamma)$.
2. Initialize:

$M = I + hc^T$, $\bar{x} = \hat{x}$, $\bar{B} = \hat{B}$.

3. Test for optimality:

If $c^T\bar{x} - \bar{B} \leq \epsilon$, stop.

4. Compute direction:

Define $\bar{y} = \bar{x} + h(c^T\bar{x} - \bar{B})$, $\bar{A} = AM^{-1}\bar{Y}$, $\bar{c} = \bar{Y}c$, $\bar{b} = b - (\bar{B}/(1 + c^Th))Ah$, $\bar{\Delta} = (\bar{c}^Te - \bar{B})/(1 + c^Th) = c^T\bar{x} - \bar{B}$, $\bar{g} = (q/((1 + c^Th)\bar{\Delta}))\bar{c} - e$, $\bar{d} = [I - \bar{A}^T(\bar{A}\bar{A}^T)^{-1}\bar{A}]\bar{g}$.
If $\|\bar{d}\| \geq \gamma$, go to Step 5. Otherwise, go to Step 6.

5. Primal Step:

Update $\bar{x} \leftarrow \bar{x} - \alpha M^{-1}\bar{Y}d$, where α is determined by a line search of the potential function $F(., \bar{B})$. Go to Step 3.

6. Dual Step:

Define $\bar{t} = (\bar{\Delta}/q)\bar{Y}^{-1}(e + \bar{d})$, $\bar{\lambda} = (\bar{\Delta}/q)(\bar{A}\bar{A}^T)^{-1}\bar{A}\bar{g}$. Set $\bar{s} = (1/(1 - h^T\bar{t}))\bar{t}$, $\bar{\pi} = (1/(1 - h^T\bar{t}))\bar{\lambda}$, $\bar{B} = b^T\bar{\pi}$. Go to Step 3.

The primal step updates the solution to the standard form linear program (PM(B)) and then transforms back to the original variables. In the dual step, the variables $\bar{\lambda}$ are feasible in the dual linear program to (PM(B)) with slacks $\bar{t}$, and the variables $\bar{\pi}$ are feasible in the dual problem to (P) with slacks $\bar{s}$. The choice of $\bar{B}$ in the dual update ensures that $\bar{x}$ is strictly feasible in (P($\bar{B}$)).

Freund shows that when $q = n + \sqrt{n}$, the algorithm converges in $O(nL)$ iterations, where L is the required accuracy of the solution, under certain assumptions concerning (P), the vector h, and the parameters. One of the assumptions regards the size of the dual feasible region, which could potentially be restrictive. This assumption places limits on the size of the shift variables h and appears to be required theoretically. It does appear necessary in practice to keep h small but not as small as indicated theoretically (see §5).

2.2. Using Freund's method in a column generation algorithm. We present a column generation algorithm that is based on Freund's algorithm. Consider the dual pair of linear programs

$$(P_{01}) \qquad \begin{array}{lll} \min & {c^0}^T x^0 + {c^1}^T x^1 & \\ \text{s.t.} & A^0 x^0 + & A^1 x^1 = b, \\ & x^0, & x^1 \geq 0 \end{array}$$

and

$$(D_{01}) \qquad \begin{array}{ll} \max & b^T\pi \\ \text{s.t.} & {A^0}^T\pi \leq c^0, \\ & {A^1}^T\pi \leq c^1. \end{array}$$

The matrix A^0 is $m \times n_0$, A^1 is $m \times n_1$, and the vectors x^0, x^1, π, c^0, c^1, and b are dimensioned appropriately. The understanding here is that the set of dual constraints

${A^0}^T\pi \leq c^0$ is easy to handle and the set of constraints ${A^1}^T\pi \leq c^1$ is harder, in that these constraints either make the linear algebra more difficult or they are not available explicitly and need to be obtained from an oracle. This is the case, for example, in a cutting plane method for integer programming, where the columns of A^0 constitute our original relaxation and the columns of A^1 are generated by using separation routines. It is also useful to use a column-generation method when the total number of columns is far larger than the number of rows; in this case, we start with a subset of the columns (A^0) and add extra columns (from A^1) as necessary, depending on their reduced costs. Thus, we originally use just the first set of constraints, adding elements from the second set as necessary.

We let n denote the number of primal variables in the current relaxation. When adding an extra constraint to the dual problem, a primal feasible solution can be obtained by giving the additional primal variable x_i value zero, but the resulting point is not interior. To overcome this difficulty, we replace the constraint $x_i \geq 0$ by the constraint $x_i + h_i(c^T x + c_i x_i - B) \geq 0$, where x_i is the added variable, B is the current lower bound on the optimal value of (P_{01}), h_i is a positive number, and x was the vector of primal variables prior to the introduction of x_i with corresponding objective function c. The initial relaxation of (P_{01}) is

$$(P_0) \qquad \begin{array}{ll} \min & {c^0}^T x^0 \\ \text{s.t.} & A^0 x^0 = b, \\ & x^0 \geq 0, \end{array}$$

with the corresponding dual problem being denoted (D_0).

The complete algorithm is as follows. We assume we know a strictly positive point $\hat{x}^0$ satisfying $A^0\hat{x}^0 = b$ and a lower bound $\hat{B}$ on the optimal value of (P_{01}).

ALGORITHM 2.2.

1. Input data $(A^0, A^1, b, c^0, c^1, \hat{x}^0, \hat{B}, \epsilon, q, \gamma)$.
2. Initialize:

$M = I$, $\bar{x} = \hat{x}^0$, $A = A^0$, $c = c^0$, $h = 0$, $\bar{B} = \hat{B}$.

3. Test for optimality:

If $c^T\bar{x} - \bar{B} \leq \epsilon$, stop.

4. Compute direction:

Define $\bar{y} = \bar{x} + h(c^T\bar{x} - \bar{B})$, $\bar{A} = AM^{-1}\bar{Y}$, $\bar{c} = \bar{Y}c$, $\bar{b} = b - (\bar{B}/(1 + c^T h))Ah$, $\bar{\Delta} = (\bar{c}^T e - \bar{B})/(1 + c^T h) = c^T\bar{x} - \bar{B}$, $\bar{g} = (q/((1 + c^T h)\bar{\Delta}))\bar{c} - e$, $\bar{d} = [I - \bar{A}^T(\bar{A}\bar{A}^T)^{-1}\bar{A}]\bar{g}$.

If $\|\bar{d}\| \geq \gamma$, go to Step 5. Otherwise, go to Step 6.

5. Primal Step:

Update $\bar{x} \leftarrow \bar{x} - \alpha M^{-1}\bar{Y}d$, where α is determined by a line search of the potential function $F(., \bar{B})$. Go to Step 3.

6. Dual Step:

(i) Update dual solution:

Define $\bar{t} = (\bar{\Delta}/q)\bar{Y}^{-1}(e + \bar{d})$, $\bar{\lambda} = (\bar{\Delta}/q)(\bar{A}\bar{A}^T)^{-1}\bar{A}\bar{g}$. Set $\tilde{s} = (1/(1 - h^T\bar{t}))\bar{t}$, $\tilde{\pi} = (1/(1 - h^T\bar{t}))\bar{\lambda}$.

(ii) Check feasibility in (D_{01}):

If ${A^1}^T\tilde{\pi} \leq c^1$ then

$\bar{\pi} = \tilde{\pi}$, $\bar{s} = \tilde{s}$, $\bar{B} = b^T\bar{\pi}$, go to Step 3;

else

Add a subset of the columns of A^1 to the columns of A, update c appropriately, set the additional components of x to zero, give the additional com-

ponents of h some positive value, update $M = I + hc^T$, update q, go to Step 3.

Unlike Algorithm 2.1, Algorithm 2.2 builds up the set of columns in the matrix A. The vector h only has positive entries for the variables that are added after starting. The column generation is accomplished in Step 6. If the generated dual solution $\bar{\pi}$ is feasible in (D_{01}), there is no need to modify the set of columns currently being used. However, if $\bar{\pi}$ satisfies $A^T\bar{\pi} \leq c$ but is infeasible in (D_{01}) then additional columns that correspond to some of the constraints violated by $\bar{\pi}$ can be added to A. There are many ways to choose the additional columns; for example, we could add all violated constraints, or we could sort them and add the most violated constraints, or we could add a subset which keeps the calculation of the necessary projections relatively easy. Note that $\bar{B}$ is always a valid lower bound on the optimal value of (P_{01}) since it is only updated when we find a point $\bar{\pi}$ that is feasible in (D_{01}). The parameter q is updated to ensure that it remains larger than the number n of primal variables; for example, we could take $q = n + \sqrt{n}$. The entries of h that were already set are left at their current values; for columns that are introduced, h_i is chosen so that the assumptions in the next subsection are satisfied.

2.3. Polynomial convergence. Under certain assumptions, Algorithm 2.2 is polynomially convergent in the total size of the problem (P_{01}). We assume that all data are integer. Let L be the size of the linear program. Here, we use the definition of Schrijver [15], so 2^L is approximately the sum of the product $m(n_0 + n_1)$ and the absolute values of the entries of the objective functions c^0 and c^1, the right-hand side b, and the constraint matrices A^0 and A^1. (This is similar to the definition in Gonzaga [8] and elsewhere but not identical. However, the differences are not significant.) Then in order to solve the linear program (P_{01}) it is sufficient to obtain a feasible solution having duality gap less than 2^{-L}.

Assumption 1. The rows of A^0 have full rank.

Assumption 2. The sets of optimal solutions to (P_0) and (D_{01}) are nonempty; the sets of optimal solutions to (P_{01}) and (D_0) are bounded. (Note that this implies that the set of optimal solutions to any relaxation generated by the algorithm must be both nonempty and bounded.)

Assumption 3. A point $\hat{x}^0 > 0$ that is feasible in (P_0) and a lower bound $\hat{B}$ on the optimal value of (P_{01}) are known. Further, ${c^0}^T\hat{x}^0 - \hat{B} \leq 2^L$.

Assumption 4. The feasible region of (D_0) is bounded.

Assumption 5. The shift vector h satisfies $h^T(c^1 - {A^1}^T\pi) \leq 1/(9\sqrt{n_0 + n_1})$ for all feasible solutions π to (D_0).

Assumption 6. $1 + \sum_{i \in I} h_i c_i^1 \neq 0$ for any choice of index set I.

As we now argue, the assumptions ensure that Freund's assumptions hold for each relaxation.

By Theorem 10.3 in Schrijver [15], the sum of the entries in $c - A^T y$ is no bigger than 4×2^L for y feasible. It follows that taking

$$h_j = \frac{1}{36(\sqrt{n_0 + n_1})2^L} \tag{1}$$

will ensure that Assumptions 5 and 6 hold.

If we update $q = n + \sqrt{n}$ then all of Freund's assumptions hold for each relaxation. Thus, we can shrink the duality gap by a factor of 2^L in $O(nL)$ iterations. However, it is not necessarily true that the duality gap after one column-generation step is less

than the duality gap immediately after the previous one. In order to prove polynomial convergence, we need to argue that the potential function remains bounded by a polynomial when columns are added. It will then follow that each relaxation can be solved in polynomial time. Note that the increase in the potential function when column $n+1$ is added is

$$\left((n+1+\sqrt{n+1})\ln(c^Tx-B)-\sum_{j=1}^{n+1}\ln(x_j+h_j(c^Tx-B))\right)$$
$$-\left((n+\sqrt{n})\ln(c^Tx-B)-\sum_{j=1}^{n}\ln(x_j+h_j(c^Tx-B))\right)$$
$$=(\sqrt{n+1}-\sqrt{n})\ln(c^Tx-B)-\ln(h_j),$$

since $x_{n+1}=0$. Since the potential function decreases by $1/12$ whenever a primal or a (noncolumn-generating) dual step is taken, it follows from Assumptions 2 and 3 and standard results that $\ln(c^Tx-B)$ is $O(L)$ at each iteration. Therefore, the potential function $F(x,B)$ is still polynomial in the size of the problem after a column is added, so the number of iterations between column generation steps is polynomial, and therefore the complete algorithm is polynomial.

In fact, we can show more than this. The total increase in the potential function due to all the column generation steps is no more than $O(\sqrt{n_0+n_1}L+n_1L)$ if h_j is selected as in (1). Now the initial value of the potential function is $O(n_0L)$. Thus the total necessary decrease in the potential function is $O((n_0+n_1)L)$. Since each noncolumn-generation iteration decreases the potential function by a constant, it follows that the algorithm converges to a point with duality gap smaller than 2^{-L} in $O((n_0+n_1)L)$ iterations. This is the same order of convergence as that displayed for the shifted barrier method in Freund [4].

3. The first time a variable is added.

3.1. The initial primal direction. In this section, we consider the direction obtained when the first extra column is generated. Thus, our current relaxation is (P_0) and the point $\bar{x}^0$ is a feasible interior point for this problem. After adding the variable x_i, we obtain the problem

$$(P_{0i})\qquad \begin{array}{lll} \min & {c^0}^Tx^0+ & c_ix_i \\ \text{s.t.} & A^0x^0+ & a_ix_i=b, \\ & x^0\geq 0, & x_i\geq 0, \end{array}$$

which we relax to

$$(P_{0i}(B))\qquad \begin{array}{lll} \min & {c^0}^Tx^0+ & c_ix_i \\ \text{s.t.} & A^0x^0+ & a_ix_i \qquad =b, \\ & x^0\geq 0, x_i+h_i({c^0}^Tx^0+c_ix_i-B)\geq 0. & \end{array}$$

Here, a_i and c_i are the parts of A^1 and c^1, respectively, corresponding to the variable x_i; h_i is a positive constant. The point $x^0=\bar{x}^0$, $x_i=0$ is a feasible point for this problem. After the change of variables $y=x^0$, $y_i=x_i+h_i({c^0}^Tx^0+c_ix_i-B)$, the

problem

$$(\mathrm{PM}_{0i}(\mathrm{B})) \qquad \begin{array}{ll} \min & rc^{0T}y + rc_i y_i \\ \text{s.t.} & (A^0 - h_i r a_i c^{0T})y + r a_i y_i = b - rBh_i a_i, \\ & y, y_i \geq 0 \end{array}$$

is obtained. Here,

$$r = \frac{1}{1 + c_i h_i}. \tag{2}$$

The point $\bar{y} = \bar{x}^0$, $\bar{y}_i = h_i(c^{0T}\bar{x}^0 - B)$ is a strictly positive feasible point for $(\mathrm{PM}_{0i}(\mathrm{B}))$.

Define

$$\bar{A} := [(A^0 - r h_i a_i c^{0T})\bar{Y} \mid (r a_i)\bar{y}_i]. \tag{3}$$

We need to find the projection of a vector onto the null space of $\bar{A}$. This requires finding $(\bar{A}\bar{A}^T)^{-1}$. The following lemma follows from direct calculation.

LEMMA 3.1. *The matrix product*

$$\begin{aligned} \bar{A}\bar{A}^T = {} & A^0\bar{Y}^2A^{0T} - h_i r A^0\bar{Y}^2c^0a_i^T - h_i r a^i c^{0T}\bar{Y}^2A^{0T} \\ & + r^2(h_i^2 c^{0T}\bar{Y}^2c^0 + \bar{y}_i{}^2)a^i a^{iT}. \end{aligned} \tag{4}$$

In order to simplify notation, we define the following quantities:

$$Q = A^0\bar{Y}^2A^{0T}, \tag{5}$$

$$k = r\sqrt{h_i^2 c^{0T}\bar{Y}^2c^0 + \bar{y}_i{}^2}, \tag{6}$$

$$l = \frac{h_i}{\sqrt{h_i^2 c^{0T}\bar{Y}^2c^0 + \bar{y}_i{}^2}}, \tag{7}$$

$$u = ka_i, \tag{8}$$

$$v = lA^0\bar{Y}^2c^0, \tag{9}$$

so

$$\begin{aligned} \bar{A}\bar{A}^T &= Q + uu^T - uv^T - vu^T \\ &= Q + (u - v)(u - v)^T - vv^T, \end{aligned} \tag{10}$$

a rank 2 modification to Q. Note that

$$kl = rh_i. \tag{11}$$

The Sherman–Morrison formula can be used to express $(\bar{A}\bar{A}^T)^{-1}$ in terms of Q^{-1}. We obtain the following rank 2 update to Q^{-1}:

$$\begin{aligned} (\bar{A}\bar{A}^T)^{-1} &= Q^{-1} + \frac{1}{\zeta}Q^{-1}(\eta uu^T + \mu uv^T + \mu vu^T + \nu vv^T)Q^{-1} \\ &= Q^{-1} + \frac{1}{\nu\zeta}Q^{-1}[(\mu u + \nu v)(\mu u + \nu v)^T]Q^{-1} \\ &\quad - \frac{1}{\nu}Q^{-1}uu^TQ^{-1}, \end{aligned} \tag{12}$$

where

$$\eta = v^T Q^{-1} v - 1, \tag{13}$$

$$\mu = 1 - u^T Q^{-1} v, \tag{14}$$

$$\nu = u^T Q^{-1} u, \tag{15}$$

$$\zeta = 1 + u^T Q^{-1} u - 2v^T Q^{-1} u - u^T Q^{-1} u v^T Q^{-1} v + (u^T Q^{-1} v)^2, \tag{16}$$

$$= (1 - u^T Q^{-1} v)^2 + u^T Q^{-1} u (1 - v^T Q^{-1} v), \tag{17}$$

$$= \mu^2 - \eta\nu. \tag{18}$$

We are now in a position to find the projection of a vector onto the null space of the matrix $\bar{A}$. Before doing that, we consider the signs of η and ζ.

LEMMA 3.2. *If either $\bar{y}_i > 0$ or $\bar{Y} c^0$ is not in the row space of $A^0 \bar{Y}$ then $\eta < 0$ and $\zeta > 0$.*

Proof. From (5), (7), (9), and (13), we have

$$\begin{aligned} \eta &= v^T Q^{-1} v - 1 \\ &= \frac{h_i^2 c^{0T} \bar{Y}^2 A^{0T} (A^0 \bar{Y}^2 A^{0T})^{-1} A^0 \bar{Y}^2 c^0}{h_i^2 c^{0T} \bar{Y}^2 c^0 + \bar{y}_i{}^2} - 1 \\ &< 0 \quad \text{from the hypothesis of the lemma.} \end{aligned}$$

In addition, from (14) and (15), $\nu \geq 0$, and if $\nu = 0$ then $\mu = 1$. It follows from (18) that $\zeta > 0$. □

Notice that the conditions of the lemma are easily satisfied; in particular, if $h_i > 0$ then $\bar{y}_i > 0$, and if $\bar{x}^0$ is not optimal for the problem (P_0) then $\bar{Y} c^0$ is not in the row space of $A^0 \bar{Y}$. From the definition of Algorithm 2.2, $h_i > 0$, so $1/\zeta$ is defined.

THEOREM 3.3. *The projection (p, p_i) of a vector (s, s_i) onto the null space of $\bar{A}$ is given by*

$$\begin{aligned} p = s &- \bar{Y} A^{0T} Q^{-1} A^0 \bar{Y} s + h_i \rho_s (\bar{Y} c^0 - \bar{Y} A^{0T} Q^{-1} A^0 \bar{Y}^2 c^0) \\ &- \tau_s \bar{Y} A^{0T} Q^{-1} a_i, \end{aligned} \tag{19}$$

$$p_i = s_i - y_i \rho_s, \tag{20}$$

where

$$\begin{aligned} \rho_s = \frac{r}{\zeta} \Big[& a_i^T Q^{-1} A^0 \bar{Y} s - k l a_i^T Q^{-1} A^0 \bar{Y} s a_i^T Q^{-1} A^0 \bar{Y} c^0 \\ & + k l a_i^T Q^{-1} a_i c^{0T} \bar{Y}^2 A^{0T} Q^{-1} A^0 \bar{Y} s + \chi_{s_i} a_i^T Q^{-1} a_i \Big], \end{aligned} \tag{21}$$

$$\tau_s = \frac{1}{\zeta} \Big[\mu k l c^{0T} \bar{Y}^2 A^{0T} Q^{-1} A^0 \bar{Y} s, + \nu k^2 a_i^T Q^{-1} A^0 \bar{Y} s + \chi_{s_i} \mu \Big], \tag{22}$$

$$\chi_{s_i} = r(y_i s_i - h_i c^{0T} \bar{Y} s). \tag{23}$$

We prove this theorem in §3.3. The direction p is a linear combination of three directions, namely, the projection of s onto the null space of $A^0 \bar{Y}$, the projection of $\bar{Y} c^0$ onto this null space, and a vector in the row space of $A^0 \bar{Y}$. The vector in the row space solves the least squares problem $\min\{x^T x : A^0 \bar{Y} x = a_i\}$.

From Step 4 of Algorithm 2.2, it is necessary to calculate the projection onto the null space of $\bar{A}$ of e and of a scaled version of the objective function. Now

$$\begin{pmatrix} \bar{c} \\ \bar{c}_i \end{pmatrix} = \begin{pmatrix} \bar{Y}c^0 \\ \bar{y}_i c_i \end{pmatrix}. \tag{24}$$

Define

$$\tilde{c} = \frac{qr}{\bar{\Delta}}\bar{c}, \qquad \tilde{c}_i = \frac{qr}{\bar{\Delta}}\bar{c}_i. \tag{25}$$

It follows that the scaled primal direction is $P^{\bar{A}}(\tilde{c}-e, \tilde{c}_i - 1)$. The next theorem is then a direct consequence of Theorem 3.3.

THEOREM 3.4. *The primal direction* $(\bar{d}, \bar{d}_i)$ *given in Algorithm* 2.2, *Step* 4 *is equal to* $P^{\bar{A}}(\tilde{c}-e, \tilde{c}_i - 1)$ *and is given by*

$$\bar{d} = -P^{A^0\bar{Y}}e + \left(\frac{qr}{\bar{\Delta}} + h_i\rho_{\tilde{c}-e}\right)P^{A^0\bar{Y}}\bar{Y}c^0 - \tau_{\tilde{c}-e}\bar{Y}A^{0^T}Q^{-1}a_i, \tag{26}$$

$$\bar{d}_i = \frac{qr}{\bar{\Delta}}\bar{y}_i c_i - 1 - y_i\rho_{\tilde{c}-e} \tag{27}$$

with ρ_s *and* τ_s *as defined in Theorem* 3.3.

Hence, the direction in the original variables is a combination of the *descent direction* $-P^{A^0\bar{Y}}\bar{Y}c^0$, the *centering direction* $P^{A^0\bar{Y}}e$, and the direction $\bar{Y}A^{0^T}Q^{-1}a_i$. Every primal interior point method for linear programming uses a combination of the descent direction and the centering direction; see Hertog and Roos [11] for a survey. The direction $\bar{Y}A^{0^T}Q^{-1}a_i$ was derived in [13], also in the context of a column-generation algorithm. It was shown in [13] that this direction is obtained when the vector $(0,1)$ is projected onto the null space of a scaled version of the constraint matrix for (P_{0i}) and that a strictly feasible point for (P_{0i}) can be obtained by moving in this direction.

3.2. The change in x_i. Theorem 3.4 gives the direction followed in the space of the scaled variables $\bar{Y}^{-1}y$ and $\bar{y}_i^{-1}y_i$. The variables x^0 are the same as the variables y, so their direction follows from the theorem. In this subsection, we calculate the direction followed by the variable x_i. When setting up the problem $(\mathrm{PM}_{0i}(\mathrm{B}))$, we made the following change of variables:

$$y_i = x_i + h_i(c^{0^T}x^0 + c_i x_i - B). \tag{28}$$

It follows that

$$x_i = r(y_i - h_i(c^{0^T}x^0 - B)), \tag{29}$$

with r as given in (2).

THEOREM 3.5. *Immediately after adding the first additional variable* x_i, *the primal direction is the direction*

$$\begin{pmatrix} \delta x^0 \\ \delta x_i \end{pmatrix} = -\begin{pmatrix} \bar{Y}\bar{d} \\ \tau_{\tilde{c}-e} \end{pmatrix}, \tag{30}$$

where $\bar{d}$ *and* τ *are as in Theorems* 3.4 *and* 3.3.

Proof. The value of δx^0 follows from Theorem 3.4 and the fact that $x^0 = y$. We must have $A\delta x^0 + a_i\delta x_i = 0$. Now from (26),

$$A\delta x^0 = \tau_{\tilde{c}-e}a_i, \tag{31}$$

so $\delta x_i = -\tau_{\tilde{c}-e}$. □

3.3. Proof of Theorem 2. The theorem is proved using a sequence of lemmas. The first of these shows that the projection is a combination of the vectors indicated, and the subsequent lemmas obtain the constants ρ_s and τ_s.

LEMMA 3.6. *Define p, p_i, s, s_i, and χ_{s_i} as in Theorem* 3.3. *Then p is a combination of the vectors $s - \bar{Y}A^{0T}Q^{-1}A^0\bar{Y}s$, $\bar{Y}c^0$, $\bar{Y}A^{0T}Q^{-1}A^0\bar{Y}^2c^0$, and $\bar{Y}A^{0T}Q^{-1}a_i$.*

Proof. The projection is given by

$$\begin{pmatrix} p \\ p_i \end{pmatrix} = \begin{pmatrix} s \\ s_i \end{pmatrix} - \bar{A}^T(\bar{A}\bar{A}^T)^{-1}\bar{A}\begin{pmatrix} s \\ s_i \end{pmatrix}.$$

Now

$$\bar{A}\begin{pmatrix} s \\ s_i \end{pmatrix} = A^0\bar{Y}s + \chi_{s_i}a_i, \tag{32}$$

using (3) and (23). It follows from (8), (9), and (12) that

$$\begin{aligned} p &= s - \bar{Y}(A^{0T} - rh_ic^0a_i^T)(\bar{A}\bar{A}^T)^{-1}(A^0\bar{Y}s + \chi_{s_i}a_i) \\ &= s - \bar{Y}A^{0T}Q^{-1}(A^0\bar{Y}s + \chi_{s_i}a_i) \\ &\quad - \frac{1}{\zeta}\bar{Y}A^{0T}Q^{-1}(\eta uu^T + \mu uv^T + \mu vu^T + \nu vv^T)Q^{-1}(A^0\bar{Y}s + \chi_{s_i}a_i) \\ &\quad + rh_i\bar{Y}c^0a_i^T(\bar{A}\bar{A}^T)^{-1}(A^0\bar{Y}s + \chi_{s_i}a_i) \\ &= s - \bar{Y}A^{0T}Q^{-1}A^0\bar{Y}s \\ &\quad + rh_i\bar{Y}c^0a_i^T(\bar{A}\bar{A}^T)^{-1}(A^0\bar{Y}s + \chi_{s_i}a_i) \\ &\quad - \bar{Y}A^{0T}Q^{-1}a_i\left(\chi_{s_i} + \frac{k}{\zeta}(\eta u^T + \mu v^T)Q^{-1}(A^0\bar{Y}s + \chi_{s_i}a_i)\right) \\ &\quad - \bar{Y}A^{0T}Q^{-1}A^0\bar{Y}^2c^0\frac{l}{\zeta}(\mu u^T + \nu v^T)Q^{-1}(A^0\bar{Y}s + \chi_{s_i}a_i). \end{aligned} \tag{33}$$

The result follows. □

The following lemma is proved similarly.

LEMMA 3.7. *Define p, p_i, s, s_i, and χ_{s_i} as in Theorem* 3.3. *Then*

$$p_i = s_i - ry_ia_i^T(\bar{A}\bar{A}^T)^{-1}(A^0\bar{Y}s + \chi_{s_i}a_i). \tag{34}$$

We now relate the multipliers derived in the proof of Lemma 3.6 to the scalars ρ_s and τ_s given in Theorem 3.3.

LEMMA 3.8. *The quantity $ra_i^T(\bar{A}\bar{A}^T)^{-1}(A^0\bar{Y}s + \chi_{s_i}a_i)$ is equal to ρ_s, with ρ_s given in* (21).

Proof. From (8), (9), (12), (13)–(15), and (18),

$$\begin{aligned} &\zeta\left(a_i^T(\bar{A}\bar{A}^T)^{-1}(A^0\bar{Y}s + \chi_{s_i}a_i)\right) \\ &\quad = \zeta a_i^TQ^{-1}A^0\bar{Y}s + \zeta\chi_{s_i}a_i^TQ^{-1}a_i \\ &\qquad + a_i^TQ^{-1}(\eta uu^T + \mu uv^T + \mu vu^T + \nu vv^T)Q^{-1}A^0\bar{Y}s \\ &\qquad + \chi_{s_i}a_i^TQ^{-1}(\eta uu^T + \mu uv^T + \mu vu^T + \nu vv^T)Q^{-1}a_i \\ &\quad = a_i^TQ^{-1}A^0\bar{Y}s(\zeta + \eta k^2a_i^TQ^{-1}a_i + \mu kla_i^TQ^{-1}A^0\bar{Y}^2c^0) \\ &\qquad + lc^{0T}\bar{Y}^2A^{0T}Q^{-1}A^0\bar{Y}s(\mu ka_i^TQ^{-1}a_i + \nu la_i^TQ^{-1}A^0\bar{Y}^2c^0) \end{aligned}$$

$$
\begin{aligned}
&+ \frac{\chi_{s_i}}{k^2}(\zeta u^T Q^{-1}u + \eta(u^T Q^{-1}u)^2 + 2\mu u^T Q^{-1}uu^T Q^{-1}v + \nu(u^T Q^{-1}v)^2)\\
&= a_i^T Q^{-1}A^0\bar{Y}s(\mu^2 - \eta\nu + \eta\nu + \mu(1-\mu))\\
&\quad + \frac{l}{k}c^{0T}\bar{Y}^2A^{0T}Q^{-1}A^0\bar{Y}s(\mu\nu + \nu(1-\mu))\\
&\quad + \frac{\chi_{s_i}}{k^2}((\mu^2 - \eta\nu)\nu + \eta\nu^2 + 2\mu\nu(1-\mu) + \nu(1-\mu)^2)\\
&= a_i^T Q^{-1}A^0\bar{Y}s(1 - kla_i^T Q^{-1}A^0\bar{Y}^2c^0) + klc^{0T}\bar{Y}^2A^{0T}Q^{-1}A^0\bar{Y}sa_i^T Q^{-1}a_i\\
&\quad + \chi_{s_i}a_i^T Q^{-1}a_i\\
&= \frac{\zeta\rho_s}{r}.
\end{aligned}
$$

The result follows. □

LEMMA 3.9. *The quantity* $(l/\zeta)(\mu u^T + \nu v^T)Q^{-1}(A^0\bar{Y}s + \chi_{s_i}a_i)$ *is equal to* $h_i\rho_s$, *with* ρ_s *given in* (21).

Proof. From (8), (9), (13)–(15), (11), and (2),

$$
\begin{aligned}
&\zeta\frac{l}{\zeta}(\mu u^T + \nu v^T)Q^{-1}(A^0\bar{Y}s + \chi_{s_i}a_i)\\
&\quad = kl\mu a_i^T Q^{-1}A^0\bar{Y}s + l^2\nu c^{0T}\bar{Y}^2A^{0T}Q^{-1}A^0\bar{Y}s\\
&\qquad + \frac{l\chi_{s_i}}{k}(\mu u^T Q^{-1}u + \nu v^T Q^{-1}u)\\
&\quad = kla_i^T Q^{-1}A^0\bar{Y}s(1 - kla_i^T Q^{-1}A^0\bar{Y}^2c^0) + (kl)^2c^{0T}\bar{Y}^2A^{0T}Q^{-1}A^0\bar{Y}sa_i^T Q^{-1}a_i\\
&\qquad + kl\chi_{s_i}a_i^T Q^{-1}a_i\\
&\quad = \zeta h_i\rho_s.
\end{aligned}
$$

The result follows. □

LEMMA 3.10. *The quantity* $\chi_{s_i} + (k/\zeta)(\eta u^T + \mu v^T)Q^{-1}(A^0\bar{Y}s + \chi_{s_i}a_i)$ *is equal to* τ_s, *with* τ_s *as given in* (22).

Proof. From (8), (9), (13)–(15), and (18),

$$
\begin{aligned}
&\zeta\left(\chi_{s_i} + \frac{k}{\zeta}(\eta u^T + \mu v^T)Q^{-1}(A^0\bar{Y}s + \chi_{s_i}a_i)\right)\\
&\quad = \chi_{s_i}(\zeta + \eta u^T Q^{-1}u + \mu v^T Q^{-1}u)\\
&\qquad + \eta k^2 a_i^T Q^{-1}A^0\bar{Y}s + \mu klc^{0T}\bar{Y}^2A^{0T}Q^{-1}A^0\bar{Y}s\\
&\quad = \chi_{s_i}(\zeta + \eta\nu + \mu(1-\mu))\\
&\qquad + \eta k^2 a_i^T Q^{-1}A^0\bar{Y}s + \mu klc^{0T}\bar{Y}^2A^{0T}Q^{-1}A^0\bar{Y}s\\
&\quad = \mu\chi_{s_i} + \eta k^2 a_i^T Q^{-1}A^0\bar{Y}s + \mu klc^{0T}\bar{Y}^2A^{0T}Q^{-1}A^0\bar{Y}s.
\end{aligned}
$$

The result follows. □

Theorem 3.3 follows from (33).

4. Subsequent additional variables. In the previous section, we considered what happens when the first column is generated. In this section, we consider the more general case when additional columns are generated. Thus, we assume that the current linear programming problem is

$$
(\mathrm{P_G(B)}) \qquad \begin{aligned} &\min && c^T x\\ &\text{s.t.} && Ax = b,\\ & && x + h(c^T x - B) \geq 0, \end{aligned}
$$

where B is the current lower bound on the optimal value of P. We add variable x_i, giving the problem

$$(\mathrm{P}_{\mathrm{G}i}(\mathrm{B}(\mathrm{G}))) \qquad \begin{array}{lll} \min & c^T x + & c_i x_i \\ \text{s.t.} & Ax + & a_i x_i = b, \\ & x + h(c^T x - B) \geq 0, & x_i \geq 0, \end{array}$$

which we relax to

$$(\mathrm{P}_{\mathrm{G}i}(\mathrm{B})) \qquad \begin{array}{lll} \min & c^T x + & c_i x_i \\ \text{s.t.} & Ax + & a_i x_i = b, \\ & x + h(c^T x + c_i x_i - B) \geq 0, & x_i + h_i(c^T x + c_i x_i - B) \geq 0. \end{array}$$

We then make the change of variables

$$y = x + h(c^T x + c_i x_i - B), \tag{35}$$

$$y_i = x_i + h_i(c^T x + c_i x_i - B). \tag{36}$$

Equation (35) implies

$$x = (I + hc^T)^{-1} y - (I + hc^T)^{-1} h c_i x_i + B(I + hc^T)^{-1} h. \tag{37}$$

Substituting in $(\mathrm{P}_{\mathrm{G}i}(\mathrm{B}))$ for x gives

$$(\mathrm{P}_{yi}(\check{\mathrm{B}})) \qquad \begin{array}{lll} \min & \check{c}^T y + & \check{c}_i x_i \\ \text{s.t.} & \check{A} y + & \check{a}_i x_i = \check{b}, \\ & y \geq 0, & x_i + h_i(\check{c}^T y + \check{c}_i x_i - \check{B}) \geq 0, \end{array}$$

where

$$\check{c} = (I + hc^T)^{-T} c = (1/(1 + c^T h))c, \tag{38}$$

$$\check{c}_i = c_i(1 - c^T(I + hc^T)^{-1} h) = (c_i/(1 + c^T h)), \tag{39}$$

$$\check{A} = A(I + hc^T)^{-1}, \tag{40}$$

$$\check{a}_i = a_i - c_i A(I + hc^T)^{-1} h = a_i - (c_i/(1 + c^T h))Ah, \tag{41}$$

$$\check{b} = b - BA(I + hc^T)^{-1} h = b - (B/(1 + c^T h))Ah, \tag{42}$$

$$\check{B} = B(1 - c^T(I + hc^T)^{-1} h) = (B/(1 + c^T h)). \tag{43}$$

The difference between B and $\check{B}$ is the same as the difference in objective function value between a feasible point for (P) and the corresponding feasible point for $(\mathrm{P}_{yi}(\check{\mathrm{B}}))$. This problem has exactly the same structure as the problem $(\mathrm{P}_{0i}(\mathrm{B}))$; it follows that the direction obtained in the variables y and y_i can be calculated as in Theorem 3.4 and the change in x_i can be found as in Theorem 3.5.

5. Preliminary computational results. In order to investigate the computational performance of the algorithm described in this paper, we randomly generated several transportation problems of sizes up to 2000 nodes. All computations were performed on an IBM 3090 under the AIX/370 operating system. All codes were written in Fortran and compiled with the FORTRANVS compiler. Times were measured using the CPUTIME subroutine of the Fortran run-time library. It cannot be expected that the implementation of the shifted barrier column generation algorithm will be competitive with codes designed specifically for network problems such as NETFLO [10]; the object of this testing is to show that an interior point column

generation method which uses shifted barriers will outperform an algorithm which solves the whole linear program. Thus, the set of test problems should be regarded purely as linear programming problems that have many more columns than rows and that possess a sparse structure that is somewhat real-world.

The problems we generated are of the standard form

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax = b, \\ & x \geq 0. \end{aligned}$$

Here A is of the form

$$A = \begin{pmatrix} e^T & 0 & \cdots & 0 \\ 0 & e^T & \cdots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & e^T \\ I & I & I & I \end{pmatrix},$$

having $f_s + f_d$ rows and $f_s \times f_d$ columns, where f_s (f_d) is the number of sources (destinations). The entries of c are randomly generated rational numbers between 0 and 1. The entries of b are randomly generated integers between 1 and 1000. In all the problems we generated, we chose $f_d = f_s$ and these numbers ranged between 100 and 1000. Because the matrix A has rank $f_s + f_d - 1$, we deleted the final row; thus our constraint matrix had $m = f_s + f_d - 1$ rows.

Various modifications to the algorithm presented in §2 are necessary for a practical implementation. We found that it was necessary to make several conservative choices in order for the algorithm to be numerically stable.

The following choices were made for the shifted barrier algorithm. We took $q = 2n$ and regarded the problem as solved if the relative duality gap (that is, the ratio of the duality gap to the current dual value) was smaller than 10^{-4}. We used a "big-M" procedure to obtain feasibility, with the objective function coefficient for the artificial variable set to 10^8. For the primal variables x_i in the initial relaxation, we set $h_i = 10^{-8}$, except that for the artificial variable $h_i = 0$. This choice of h is small enough that the algorithm closely approximates a standard primal projective method (see Ye [17] or Freund [3]). The initial primal point is the vector of ones, with the artificial column chosen appropriately. The vector of zeros is dual feasible; we used a heuristic to find a slightly better initial dual point that was also feasible in the whole problem. The dual variables $\bar{\pi}$ are updated by solving a one-variable linear programming problem similar to that in Todd and Burrell [16] (see also Gonzaga [7]). The calculation of the primal direction is obtained by finding projections of the three vectors h, c, and e, appropriately scaled, onto an appropriate subspace; updating $\bar{\pi}$ then requires performing a minimum ratio test involving these three vectors. (We also attempted to use various two-variable subproblems to update $\bar{\pi}$, but these produced very little improvement and were numerically difficult.) If we can improve the dual solution we do so; we always update the primal solution unless the dual update decreases the duality gap by a factor of at least two. The projections were calculated using a modified version of the Yale Sparse Matrix Package [2]. This uses the minimum degree ordering routine in combination with a Cholesky factorization and then forward and back substitution. The primal step length was .99 of the distance to the boundary.

The column generation aspects of the algorithm were implemented as follows. The columns in the initial relaxation were chosen heuristically. First, a greedy method

was used which attempted to add as few columns as possible while ensuring that the initial primal relaxation contained strictly positive feasible points. Second, we found it helpful to include also some promising columns in the initial relaxation. To determine which columns were promising, we examined the ratio $\log((b_i + b_j)/c_{ij})$, where c_{ij} is the cost of the column and b_i and b_j are the two elements of the right-hand side corresponding to the nonzeros in the column. If this ratio is small, it is unlikely that the column is in the optimal solution. We used a bucket sort to select the promising columns and picked no more than $5m$ such columns. A poor initial set of columns hurt the algorithm dramatically. We found a primal feasible point before looking for additional columns. Whenever the dual solution is updated, the algorithm looks for violated dual constraints. For variables added through column generation, the choice of h_i depends on the number of variables added at that stage: we chose $h_i = 10^{-2}/ADDVAR$, where $ADDVAR$ variables are introduced at the given stage. This choice of h_i was decided upon after experimentation with various alternatives; h_i should not be too small if the new iterate is to be somewhat centered, but taking h_i too large causes numerical problems. (Recall that in the theoretical analysis, we require (Assumption 5) that the shift vector h satisfies $h^T(c^1 - A^{1^T}\pi) \leq 1/(9\sqrt{n_0 + n_1})$ for all feasible solutions π to (D_0). This is used to place a bound on the size of $h^T\bar{t}$ in the dual step in the algorithm.) When deciding which columns to add, we bucket sorted the columns by the violation of the current dual solution $\bar{\pi}$ and then added no more than approximately $2m$ columns. After adding the columns, the dual solution is set to its previous value, which is still feasible. In what follows, we refer to this algorithm by the mnemonic *SBCG*.

We compared the shifted barrier column generation algorithm with two other shifted barrier algorithms. The first algorithm (*SBFull*) runs the shifted barrier algorithm as described above but applied to the full set of columns. Thus, *SBFull* does not use column generation. The second algorithm (*SBCGArt*) is another shifted barrier column generation algorithm. This algorithm differs from *SBCG* in the way it handles generated columns. It sets $h_i = 10^{-8}$ for all introduced variables and also introduces an artificial variable at each stage. The cost of the artificial variable is set to 10^5. All the new primal variables and the artificial variable are set to one, with the artificial column defined appropriately. At each stage the artificial variable is eliminated before searching for new columns. Usually, the Phase I procedure only takes one iteration; occasionally it takes longer when many variables are added at once. Because h_i is set so small, this algorithm approximates a simple unshifted column generation method.

5.1. Comparing column generation with no column generation. Table 1 compares the three algorithms on 15 problems. For each algorithm, exactly the same parameter settings were used for all fifteen runs. The columns headed *Iters* record the number of iterations required by a particular algorithm for a given problem. (The number of "iterations" is equivalent to the number of times the projection matrix is factored. An iteration may involve just a primal update, just a dual update, both a primal and a dual update, a dual update and then column generation, or a primal and dual update and then column generation.) The columns headed *Time* record the number of seconds (CPU time) required by a particular algorithm for a given problem in each run, calculating the Cholesky factors takes about 80 to 90% of the CPU time. The average values for the 100, 200, and 400 source problems are included in the table. As can be seen from the table, the column generation algorithms outperformed the other algorithm, with the difference increasing as the problem size increased.

One reason why *SBFull* did so poorly on some of the 400 by 400 problems is that it

TABLE 1
Comparing the three algorithms.

Problem			*SBFull*		*SBCG*		*SBCGArt*	
	f_s	f_d	Iters	Time	Iters	Time	Iters	Time
1.	100	100	32	44.4	55	10.1	58	11.2
2.	100	100	33	45.3	46	10.0	40	9.7
3.	100	100	32	44.6	34	9.7	46	14.2
4.	100	100	31	42.9	45	10.1	50	12.6
5.	100	100	30	41.5	29	8.4	34	10.5
Mean.	100	100	32	43.7	42	9.7	46	11.6
6.	200	200	40	336.3	32	58.7	36	69.1
7.	200	200	39	327.6	86	93.6	87	103.5
8.	200	200	38	320.8	47	65.7	38	53.1
9.	200	200	38	320.1	81	70.4	75	70.3
10.	200	200	40	336.6	67	75.4	68	87.9
Mean.	200	200	39	328.3	63	72.8	61	76.8
11.	400	400	45	2677.0	53	595.9	52	614.6
12.	400	400	63	3563.3	60	435.5	61	463.6
13.	400	400	43	2576.2	69	474.9	64	465.2
14.	400	400	70	3907.4	83	490.8	77	472.7
15.	400	400	59	3369.1	60	484.6	66	564.9
Mean.	400	400	56	3218.6	65	496.3	64	516.2

had difficulty in improving on the initial heuristic dual solution. These problems have 160,000 dual constraints, so slight numerical inaccuracies will make it very difficult to find a solution to the 1-variable dual problem which must be solved to update the dual solution.

The runtimes could be improved considerably by using a different routine to calculate the projections. For the full constraint matrix A, the matrices AA^T and its Cholesky factor are both 75% dense. For the column generation programs, the Cholesky factors are initially approximately between 15% and 25% dense, with the density increasing to about 25% to 30% by the end of the run. Thus, using a factorization routine that can switch from a sparse factorization to a dense one part way through a factorization would help all the codes, with the greatest absolute improvement being for *SBFull*. Nonetheless, because of the large amount of time required to calculate the full product AA^T, the greatest percentage improvement would probably be for the column generation codes.

Even if iteration counts are identical, the code *SBCGArt* should take more time than *SBCG* because of the presence of the dense artificial column each time columns are generated. *SBCGArt* is supposed to approximate a column generation algorithm that does not use shifted barriers; such an algorithm requires only two projections per iteration rather than the three used by *SBCG*. Thus the difference in run times between *SBCG* and *SBCGArt* should be discounted.

5.2. Comparing the two column generation codes. We compared the two algorithms *SBCG* and *SBCGArt* in more detail. The two algorithms are identical until columns are generated; they then generate slightly different iterates. For this purpose, we ran the two algorithms on the fifteen problems mentioned above and also on two 1000×1000 problems, using varying numbers of initial columns. See Table 2 for the results. The columns headed f_s, f_d, Iters, and Time have the same meanings as in Table 1. The column headed n_0 gives the number of columns in the initial relaxation. The columns headed Cols give the total number of columns generated. The columns

TABLE 2
Comparing the two column generation algorithms.

Problem				*SBCG*				*SBCGArt*			
	f_s	f_d	n_0	Iters	Time	Cols	Stages	Iters	Time	Cols	Stages
1.	100	100	659	55	10.1	391	6	58	11.2	381	5
			1176	28	10.4	2	2	29	11.3	2	2
2.	100	100	866	46	10.0	83	5	40	9.7	84	4
			1222	37	13.2	9	4	36	14.1	9	4
3.	100	100	1054	34	9.7	7	3	46	14.2	7	5
			1211	31	11.4	3	2	35	13.3	3	2
4.	100	100	807	45	10.1	211	4	50	12.6	211	6
			1257	27	10.6	5	3	30	12.6	5	3
5.	100	100	1084	29	8.4	5	3	34	10.5	5	3
			1228	29	10.8	2	2	31	12.2	2	2
Mean.	100	100	1056	36	10.5	72	3.4	39	12.2	71	3.6
6.	200	200	2062	36	58.0	56	4	38	62.7	56	3
			2413	32	58.7	5	3	36	69.1	5	3
			2463	31	62.5	4	3	35	73.2	4	3
7.	200	200	1538	86	93.6	694	7	87	103.5	691	8
			2101	41	68.3	53	4	42	74.4	53	4
			2504	40	62.6	42	3	36	75.5	42	3
8.	200	200	2064	47	65.7	184	5	38	53.1	185	2
			2074	39	53.6	281	3	44	65.7	280	4
			2450	33	62.6	125	3	37	72.8	124	3
9.	200	200	1366	81	70.4	625	9	75	70.3	643	8
			2125	40	72.2	316	4	43	82.3	316	4
			2522	45	83.3	27	3	36	72.6	27	3
10.	200	200	1630	67	75.4	487	6	68	87.9	489	9
			2128	45	77.6	233	4	54	99.3	233	5
			2518	41	86.1	203	3	40	91.3	203	4
Mean.	200	200	2131	47	70.0	222	4.3	47	76.9	223	4.4
11.	400	400	4366	53	595.9	69	6	52	614.6	69	6
			4464	54	610.7	67	7	51	606.8	67	6
			8981	32	801.3	2	2	34	866.1	2	2
12.	400	400	3437	60	435.5	989	5	61	463.6	999	5
			4028	51	476.9	555	5	53	526.1	555	6
			8787	32	758.5	1	1	33	789.6	1	1
13.	400	400	3467	69	474.9	832	6	64	465.2	828	6
			4265	46	467.7	689	2	50	523.5	689	3
			9052	26	655.2	1	1	29	736.4	1	1
14.	400	400	2996	83	490.8	1099	5	77	472.7	1098	5
			4108	47	458.9	488	6	53	543.2	488	6
			8876	33	781.1	1	1	34	814.5	1	1
15.	400	400	3630	60	484.6	775	5	66	564.9	768	7
			4096	44	425.3	163	3	44	437.6	163	3
			8769	34	824.2	2	2	35	863.8	2	2
Mean.	400	400	5555	48	582.8	382	3.8	49	619.2	382	4.0
16.	1000	1000	8065	141	11094.4	4037	10	143	11500.6	4009	10
17.	1000	1000	8436	138	11626.1	4917	10	124	10800.5	4879	10
Mean.	1000	1000	8250	139	11247.5	4477	10	134	11150.5	4444	10

headed Stages give the number of times columns are added to the formulation. Thus, for each problem, the total number of formulations constructed is 1 + Stages, namely, the initial formulation together with one new formulation each time columns are generated. With one exception, the runs listed in Table 1 produce the smallest value of n_0 given in Table 2; the exception is Problem 6, where the run in Table 1 gave 2413 columns in the initial relaxation.

The two algorithms perform similarly. They do add different numbers of columns and take different numbers of stages on some problems. For almost every problem, the smaller the initial number of columns, the larger the number of iterations required and the number of columns generated and stages taken. It is not necessarily true that the algorithm that takes fewer stages will take fewer iterations; if more columns are added at a particular stage, it is likely to take more iterations to complete that stage.

The runs that required the addition of many columns generally added a large number for the first few stages and then considerably fewer for the last few, with the result that the early stages take many iterations and the later ones relatively few.

When only a few columns are added (less than 10), *SBCG* outperforms *SBCGArt* in every case. Nonetheless, the two algorithms have almost identical iteration counts because of the slightly superior performance of *SBCGArt* when many columns are added. This suggests that a hybrid algorithm might well be better than either *SBCG* or *SBCGArt*: use the methodology of *SBCGArt* when many columns are added and that of *SBCG* when only a few are added.

5.3. Possible improvements. The algorithms as implemented never drop any columns. Introducing such a procedure would decrease the size of the linear programming problems considered, giving two immediate benefits: it would take less time to calculate the projections at each iteration, and it would take less iterations to solve each stage. Dropping columns when using *SBCGArt* is straightforward—just modify the artificial column. It is not so easy to modify *SBCG*, but a procedure similar to that given in [13] could be used.

It would probably be worthwhile spending more time trying to update the dual solution. Because of the number of dual constraints in the full formulation, this would help *SBCGFull*. It would also help the column generation algorithms because it is necessary to obtain a new dual solution before trying to add columns.

Because the algorithm is numerically difficult due to the small values of h_i, these implementations have been conservative in the choices of many parameters. It should be possible to speed up the algorithm by enabling some parameters to be changed dynamically depending on how the algorithm is performing.

6. Summary. We have introduced a shifted barrier column generation algorithm that requires $O(nL)$ iterations. We showed that the steps taken by this algorithm when introducing a variable are strongly related to the standard centering and affine scaling steps and also to the direction proposed in Mitchell and Todd [13].

We have implemented the algorithm. As is to be expected, the algorithm outperforms one that does not use column generation for problems that have many more columns than rows. In addition, it appears that the algorithm may well outperform a simple column generation algorithm when only a few variables are introduced. The algorithm must be implemented carefully if it is to be numerically stable. In order to significantly improve the algorithm, it is probably necessary to improve the method of updating the dual solution.

Acknowledgments. I am grateful to Eberhard Kranich for his bibliography on interior point methods, which is available through Netlib. I also thank the referees and the associate editor for their helpful comments.

REFERENCES

[1] D. S. Atkinson and P. M. Vaidya, *An analytic center based cutting plane algorithm for convex programming*, Technical report, Department of Mathematics, University of Illinois at Urbana-Champaign, June 1992.

[2] S. C. Eisenstat, M. C. Gurshy, M. H. Schultz, and A. H. Sherman, *The Yale sparse matrix package, I. The symmetric codes*, Internat. J. Numer. Methods in Engrg., 18 (1982), pp. 1145–1151.

[3] R. M. Freund, *Polynomial algorithms for linear programming based only on primal scaling and projected gradients of a potential function*, Math. Programming, 51 (1991), pp. 203–222.

[4] ———, *A potential-function reduction algorithm for solving a linear program directly from an infeasible "warm start,"* Math. Programming, 52 (1991), pp. 441–466.
[5] J.-L. GOFFIN, A. HAURIE, AND J.-P. VIAL, *Decomposition and nondifferentiable optimization with the projective algorithm*, Management Sci., 38 (1992), pp. 284–302.
[6] J. L. GOFFIN AND J. P. VIAL, *Cutting planes and column generation techniques with the projective algorithm*, J. Optim. Theory Appl., 65 (1990), pp. 409–429.
[7] C. C. GONZAGA, *On lower bound updates in primal potential reduction methods for linear programming*, Math. Programming, 52 (1991), pp. 415–428.
[8] ———, *Path following methods for linear programming*, SIAM Rev., 34 (1992), pp. 167–224.
[9] M. GRÖTSCHEL AND O. HOLLAND, *Solution of large-scale travelling salesman problems*, Math. Programming, 51 (1991), pp. 141–202.
[10] R. V. HELGASON AND J. L. KENNINGTON, *NETFLO* Computer Program, Operations Research Department, Southern Methodist University, Dallas, TX, 1985.
[11] D. HERTOG AND C. ROOS, *A survey of search directions in interior point methods for linear programming*, Math. Programming, 52 (1991), pp. 481–510.
[12] J. A. KALISKI AND Y. YE, *A decomposition variant of the potential reduction algorithm for linear programming*, Working Paper 91–11, Dept. of Management Science, College of Business Administration, University of Iowa, Iowa City, IA, 1991.
[13] J. E. MITCHELL AND M. J. TODD, *Solving combinatorial optimization problems using Karmarkar's algorithm*, Math. Programming, 56 (1992), pp. 245–284.
[14] M. W. PADBERG AND G. RINALDI, *A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems*, SIAM Rev., 33 (1991), pp. 60–100.
[15] A. SCHRIJVER, *Theory of Linear and Integer Programming*, John Wiley, Chichester, 1986.
[16] M. J. TODD AND B. P. BURRELL, *An extension of Karmarkar's algorithm for linear programming using dual variables*, Algorithmica, 1 (1986), pp. 409–424.
[17] Y. YE, *An $O(n^3L)$ potential reduction algorithm for linear programming*, Math. Programming, 50 (1991), pp. 239–258.
[18] ———, *A potential reduction algorithm allowing column generation*, SIAM J. Optim., 2 (1992), pp. 7–20.

SIAM J. OPTIMIZATION
Vol. 4, No. 2, pp. 441–453, May 1994
© 1994 Society for Industrial and Applied Mathematics
012

PREDICTOR-CORRECTOR METHODS FOR A CLASS OF LINEAR COMPLEMENTARITY PROBLEMS*

SANJAY MEHROTRA† AND ROBERT A. STUBBS†

Abstract. The Mizuno–Todd–Ye predictor-corrector method [*Mathematics of Operations Research*, to appear] for linear programming uses a Newton direction for the predictor step and a centering direction for the corrector step. This paper develops several additional $O(\sqrt{n}L)$ iteration predictor-corrector methods for the linear complementarity problem. These methods replace the Newton predictor step with directions computed from appropriately defined ball optimization problems and trust-region problems. The one-step computational requirement of methods solving the ball optimization problem is similar to the computational requirement of the method using the Newton predictor step.

Key words. linear complementarity problem, primal-dual methods, predictor-corrector methods, linear programming, quadratic programming

AMS subject classifications. 49M40, 90C05, 90C20, 90C33

1. Introduction.

We consider the linear complementarity problem (LCP):

$$Mx - y = q,$$

$$x \geq 0, \qquad y \geq 0,$$

$$x^T y = 0,$$

where $x, y \in \mathcal{R}^n$ and $M \in \mathcal{R}^{n \times n}$ is assumed to be a positive semidefinite matrix. This paper develops new methods for solving the LCP. These methods follow the predictor-corrector strategy described in Mizuno–Todd–Ye [11] for solving the linear programming problem. The methods developed in this paper differ from earlier methods in a fundamental way. Instead of using the Newton direction as a predictor direction, the new methods compute the predictor direction at each iteration by solving subproblems defined by taking ellipsoidal approximations of the feasible region. Interestingly, one of these methods, Ball Optimization Method I, reduces to the Mizuno–Todd–Ye [11] predictor-corrector method for linear programming; however, it differs from its extension for LCP given by Ye and Anstreicher [21] and Ji, Potra, and Huang [2]. Our preliminary computational results indicate that among the methods with similar computational requirements for the predictor step, Ball Optimization Method I requires considerably fewer iterations.

Kojima, Mizuno, and Yoshise [7] were first to develop an $O(\sqrt{n}L)$-iteration path-following algorithm for the LCP. In a subsequent paper Kojima, Mizuno, and Yoshise [6] developed a method with the same iteration complexity while using a potential function to measure the progress of the method. Kojima, Megiddo, and Ye [5], Ye and Pardalos [19], and Ye [16] used the potential function to solve the LCP with weaker assumptions on M. We refer the reader to Kojima et al. [8] for detailed developments of these methods.

* Received by the editors June 24, 1992; accepted for publication (in revised form) March 24, 1993. This research was supported in part by a grant from GTE Laboratories and by National Science Foundation grant CCR-9019469.

† Department of IE/MS, Technological Institute, Northwestern University, Evanston, Illinois 60208 (mehrotra@iems.nwu.edu) and (rastubbs@iems.nwu.edu).

More recently, Ye and Anstreicher [21] and Ji, Potra, and Huang [2] have given predictor-corrector methods for solving the LCP. These predictor-corrector methods are based on the Mizuno–Todd–Ye [11] predictor-corrector method for linear programming. This approach is of interest because Mehrotra [10], and Ye et al. [18] have shown that this method converges Q-quadratically for linear programming. The results for linear programming are extended by Ye and Anstreicher [21] for the LCP. They showed that the quadratic convergence results for linear programming extend to the LCP under a strict complementary assumption on the solutions of the LCP. This assumption appears to be necessary to achieve Q-quadratic convergence by the predictor-corrector methods considered by Ye and Anstreicher [21]. Ye and Anstreicher [21] also give an example of the LCP where this assumption is not satisfied, and the algorithm converges only linearly.

The methods developed in this paper consider alternative approaches for computing the predictor direction in a predictor-corrector method. The use of alternative predictor directions is motivated from the lack of quadratic convergence for the Newton direction used in Ye and Anstreicher [21] and Ji, Potra, and Huang [2]. We expect that using other directions will lead to better interior point algorithms for solving the LCP. This is particularly important because computational experience shows that the effectiveness of a method is largely determined by the progress made by the predictor direction.

The approaches developed in this paper are based on taking ellipsoidal approximations of the feasible region. These methods differ in how they approximate the feasible region and the objective function they optimize over the ellipsoidal approximation. We show that all of the methods developed in this paper solve the LCP in $O(\sqrt{n}L)$ iterations. The approach we use to analyze these methods unifies the analysis of all predictor-corrector methods.

The use of such subproblems is in the spirit of the motivations behind Karmarkar's algorithm [4] for linear programming. Kojima, Megiddo, and Ye [5] also considered a ball optimization problem to develop a potential reduction algorithm for the LCP. The affine scaling used in the methods of §4 is similar to the scaling used by Kojima, Megiddo, and Ye [5]. The primal-dual scaling used to develop the methods in §3 is also used in Todd [14] and McShane [9] to develop potential reduction algorithms for the LCP. Earlier, this scaling was used in Todd and Ye [15] to develop a method for linear programming. The use of trust-region problems has been suggested primarily in the context of methods for quadratic programming [3], [20].

This paper is organized as follows. In §2, we describe a generic predictor-corrector method. In this section we also give properties of a centering direction. This centering approach is common to all of the methods studied in this paper. Section 3 develops our first two methods that use a primal-dual scaling to define ellipsoidal approximations of the region. Section 4 develops two additional methods that use the standard affine scaling to make ellipsoidal approximations. Section 5 contains some concluding remarks.

1.1. Notation. The following notation is used throughout this paper. Let $\mathcal{F}^o = \{(x,y) \mid Mx - y = q, x > 0, y > 0\}$ and e be a vector whose every component is all 1's. The central path is defined to be the set of points $(x(\mu), y(\mu)) \in \mathcal{F}^o$ such that $x_i(\mu)y_i(\mu) = \mu$ for all i. We define $X = \operatorname{diag}(x_i)$ to be a diagonal matrix whose nonzero components are the elements of x. Similar notation is followed for other vectors. Let

$$\mathcal{N}_2(\beta) = \left\{(x,y) \in \mathcal{F}^o \mid \|Xy - \mu e\| \leq \beta\mu, \text{ where } \mu = x^T y/n\right\}.$$

$\mathcal{N}_2(\beta)$ represents a neighborhood around the central path. Additional notation is defined as needed.

2. Predictor-corrector methods, centering, and convergence. We first describe a generic framework to illustrate the predictor-corrector strategy. All of the methods studied in this paper fit this generic framework.

2.1. A generic predictor-corrector method.

Input: M, β, ϵ, $(x^0, y^0) \in \mathcal{N}_2(\beta)$.
Initialize $k = 0$
Step 0 **(Optimality Check).**
If $({x^k}^T y^k \leq \epsilon)$ **exit**; otherwise goto Step 1.
Step 1 **(Computation of a predictor direction).**
Use one of the approaches described in the following sections to compute a predictor direction (d_x, d_y).
Step 2 **(Computation of step length).**
Let $x(\theta) = x^k - \theta d_x$, $y(\theta) = y^k - \theta d_y$, and $\mu(\theta) = x(\theta)^T y(\theta)/n$,
Find the largest $\bar{\theta}$ such that $\|X(\theta)y(\theta) - \mu(\theta)e\| < 2\beta\mu(\theta)$ for $0 \leq \theta \leq \bar{\theta}$.
Step 3 **(Intermediate solution).** Let

$$\hat{x} = x^k - \bar{\theta} d_x,$$
$$\hat{y} = y^k - \bar{\theta} d_y.$$

Step 4 **(Computation of the centering direction).**
Use the method described in §2.2 to compute centering direction (p_x, p_y).
Step 5 **(Update solution).**

$$x^{k+1} = \hat{x} - p_x,$$
$$y^{k+1} = \hat{y} - p_y.$$

Step 6 **(Next Iteration).** $k = k + 1$, goto Step 0.

Exhibit 1. *A generic predictor-corrector method.*

The parameter ϵ is the optimality tolerance. $\beta = 1/6$ is taken in the analysis of all the methods developed in this paper.

2.2. Centering direction and its properties.

The use of a centering direction (or some form of centering) is instrumental in establishing global linear convergence results in interior point methods. The centering direction also plays a fundamental role in establishing convergence results for the methods studied in the next sections. We describe the computation of the centering direction and its properties below.

We fix k and let (x, y) represent (x^k, y^k) and (x^+, y^+) represent (x^{k+1}, y^{k+1}). Also, let $\mu = x^T y/n$ and $\mu^+ = {x^+}^T y^+/n$. Let $(\hat{x}, \hat{y})$ be given as in Step 3 of Exhibit 1 and $\hat{\mu} = \hat{x}^T \hat{y}/n$. The centering direction is taken to be the Newton direction for the system of equations:

$$\begin{aligned} Mx - y &= q, \\ Xy &= \hat{\mu} e \end{aligned} \tag{2.1}$$

at $(\hat{x}, \hat{y})$. It is computed by solving the system of linear equations:

$$\begin{bmatrix} M & -I \\ \hat{Y} & \hat{X} \end{bmatrix} \begin{pmatrix} p_x \\ p_y \end{pmatrix} = \begin{pmatrix} 0 \\ \hat{X}\hat{y} - \hat{\mu} e \end{pmatrix}. \tag{2.2}$$

The restriction $(\hat{x},\hat{y}) \in \mathcal{N}_2(2\beta)$ ensures quadratic convergence to the solution of (2.1). This fact leads to Lemma 2.4, which is well known [7]. We present some relevant results here for completeness. We leave the proof of the next two results for the reader.

PROPOSITION 2.1. *For* $(x,y) \in \mathcal{N}_2(\beta)$,

$$\max_i \{x_i y_i\} \leq (1+\beta)\mu, \tag{2.3}$$

$$\min_i \{x_i y_i\} \geq (1-\beta)\mu. \tag{2.4}$$

LEMMA 2.2 (Kojima, Mizuno, and Yoshise [7]). *Let* $r, s \in \mathcal{R}^n$ *and* $r^T s \geq 0$. *Then*

$$\begin{aligned} \|r\| &\leq \|r+s\|, \\ \|s\| &\leq \|r+s\|, \\ \|r\|\,\|s\| &\leq \|r+s\|^2/2, \\ \|Rs\| &\leq \|r+s\|^2/2. \end{aligned}$$

LEMMA 2.3. *Let* $(\hat{x},\hat{y}) \in \mathcal{N}_2(2\beta)$, $0 < \beta \leq 1/4$, *and* $\hat{D} = \hat{X}^{1/2}\hat{Y}^{-1/2}$. *Let* (p_x, p_y) *be computed from* (2.2). *Then*

$$\|\hat{D}p_y + \hat{D}^{-1}p_x\|^2 \leq 4\beta^2\hat{\mu}/(1-2\beta). \tag{2.5}$$

Proof. By multiplying the second set of equations in (2.2) by $(\hat{X}^{1/2}\hat{Y}^{1/2})^{-1}$, we have

$$\begin{aligned} \|\hat{D}p_y + \hat{D}^{-1}p_x\|^2 &= \left\|\hat{X}^{1/2}\hat{Y}^{1/2}e - \hat{\mu}\hat{X}^{-1/2}\hat{Y}^{-1/2}e\right\|^2 \\ &\leq \frac{\left\|\hat{X}\hat{y} - \hat{\mu}e\right\|^2}{\min(\hat{x}_i\hat{y}_i)} \leq \frac{4\beta^2\hat{\mu}^2}{(1-2\beta)\hat{\mu}} = \frac{4\beta^2\hat{\mu}}{(1-2\beta)}, \end{aligned}$$

where the second inequality uses (2.4) and $(\hat{x},\hat{y}) \in \mathcal{N}_2(2\beta)$. □

The next lemma is similar to Lemma 5 in Kojima, Mizuno, and Yoshise [7].

LEMMA 2.4. *Let* $(\hat{x},\hat{y}) \in \mathcal{N}_2(2\beta)$, $0 \leq \beta \leq 1/4$, *and* (p_x, p_y) *be computed from* (2.2). *Then* $(x^+, y^+) \in \mathcal{N}_2(\beta)$.

Proof. The second equality below uses the second set of equations in (2.2). Let $P_x = \operatorname{diag}(p_x)$. We have

$$\begin{aligned} \left\|X^+y^+ - \mu^+e\right\|^2 &= \left\|(\hat{X} - P_x)(\hat{y} - p_y) - \frac{(\hat{x}-p_x)^T(\hat{y}-p_y)}{n}e\right\|^2 \\ &= \left\|P_xp_y - \frac{p_x^Tp_y}{n}e\right\|^2 \\ &= \|P_xp_y\|^2 - \frac{(p_x^Tp_y)^2}{n} \\ &\leq \|P_xp_y\|^2. \end{aligned}$$

The first inequality below uses Lemma 2.2, and the second inequality uses Lemma 2.3. Now

$$\|P_xp_y\| = \left\|(\hat{X}^{-1/2}\hat{Y}^{1/2}P_x)(\hat{X}^{1/2}\hat{Y}^{-1/2}p_y)\right\| \leq \tfrac{1}{2}\left\|\hat{D}p_y + \hat{D}^{-1}p_x\right\|^2 \leq 2\beta^2\hat{\mu}/(1-2\beta).$$

From (2.2), the definition of $\hat{\mu}$, and $p_x^T p_y \geq 0$, we have

$$x^{+T} y^+ = (\hat{x} - p_x)^T (\hat{y} - p_y) = \hat{x}^T \hat{y} + p_x^T p_y \geq \hat{x}^T \hat{y}.$$

Therefore, $\|X^+ y^+ - \mu^+ e\| \leq 2\beta^2 \hat{\mu}/(1-2\beta) \leq 2\beta^2 \mu^+/(1-2\beta)$. It is now easy to verify that for $0 \leq \beta \leq 1/4$, $(x^+, y^+) \leq \mathcal{N}_2(\beta)$. □

LEMMA 2.5. *Let* $(\hat{x}, \hat{y}) \in \mathcal{N}_2(2\beta)$, $0 \leq \beta \leq 1/4$. *Then*

$$x^{+T} y^+ \leq \left(1 + \frac{2\beta^2}{(1-2\beta)n}\right) \hat{x}^T \hat{y}.$$

Proof. The first inequality below uses the Cauchy–Schwarz inequality, and the second inequality uses Lemmas 2.2 and 2.3. Now

$$p_x^T p_y = p_x^T \hat{D}^{-1} \hat{D} p_y \leq \|\hat{D}^{-1} p_x\| \|\hat{D} p_y\| \leq \frac{2\beta^2 \hat{x}^T \hat{y}}{(1-2\beta)n}.$$

Hence,

$$x^{+T} y^+ = (\hat{x} - p_x)^T (\hat{y} - p_y) = \hat{x}^T \hat{y} + p_x^T p_y \leq \left(1 + \frac{2\beta^2}{(1-2\beta)n}\right) \hat{x}^T \hat{y}. \quad \square$$

2.3. A common convergence theorem. The next theorem is common to all the methods developed in this paper. It is a standard theorem in the analysis of interior point methods. (See Theorem 3 and §6 in Kojima, Mizuno, and Yoshise [7].)

THEOREM 2.6. *Let* (x^0, y^0) *be a given initial feasible solution for the* LCP *and* $x^{0T} y^0 = 2^{O(L)}$, *where* L *is the input length of the* LCP. *Assume that* $(x^0, y^0) \in \mathcal{N}_2(\beta)$, $0 \leq \beta \leq 1/6$. *Now assume that an algorithm generates a sequence of iterates* $(x^k, y^k) \in \mathcal{N}_2(1/6)$ *and*

$$\frac{x^{k+1T} y^{k+1}}{x^{kT} y^k} \leq \left(1 - \frac{\gamma}{\sqrt{n}}\right),$$

where $\gamma > 0$ *is a fixed constant independent of problem data. Then this algorithm can be used to find an exact optimal solution of* LCP *in* $O(\sqrt{n}L)$ *iterations.*

Theorem (2.6) assumes that $(x^0, y^0) \in \mathcal{N}_2(\beta)$. The method described in Kojima, Mizuno, and Yoshise [7] can be used to construct an artificial problem satisfying this assumption.

3. Methods using primal-dual scaling ellipsoids. In this section, we develop two methods that are based on the primal-dual scaling. The two methods differ in the objective function that they optimize over the ellipsoidal approximation of the feasible region. Ball Optimization Method I uses a linear objective while Trust Region Method I uses a quadratic objective.

3.1. Ball Optimization Method I. Let $D = X^{1/2} Y^{-1/2}$. The method developed in this section computes the predictor direction in the generic predictor-corrector method by solving the subproblem:

$$\begin{array}{ll} \max & y^T d_x + x^T d_y \\ \text{subject to} & M d_x - d_y = 0 \\ & \left\| \begin{array}{c} D^{-1} d_x \\ D d_y \end{array} \right\|^2 \leq x^T y. \end{array} \tag{3.1}$$

Let $v = X^{1/2}Y^{1/2}e$, $g_x = D^{-1}d_x$, and $g_y = Dd_y$. Then the solution of (3.1) can be obtained by solving the problem:

$$\begin{array}{ll} \max & v^T g_x + v^T g_y \\ \text{subject to} & MDg_x - D^{-1}g_y = 0 \\ & \left\| \begin{array}{c} g_x \\ g_y \end{array} \right\|^2 \leq x^T y. \end{array} \tag{3.2}$$

The above problem maximizes a linear function over a ball in a lower dimension. Let $w = \binom{v}{v}$. The solution of (3.2) can be obtained by orthogonally projecting w onto the nullspace of $[MD| - D^{-1}]$. In particular,

$$\begin{pmatrix} g_x \\ g_y \end{pmatrix} = \rho P_B w,$$

where

$$P_B = I - B^T(BB^T)^{-1}B, \qquad B = [MD| - D^{-1}],$$

and $\rho = \sqrt{x^T y}/\left\|P_B w\right\|$, gives the solution of (3.2).

Note we have assumed that $P_B w \neq 0$. If $P_B w = 0$, then w must be in the range space of B^T, i.e., there exists a π such that

$$\begin{bmatrix} DM^T \\ -D^{-1} \end{bmatrix} \pi = w.$$

This gives $\pi = -x$ and $x^T M x = -x^T y$, which contradicts the assumption that M is a positive semidefinite matrix.

3.2. Convergence analysis for the Ball Optimization Method I. Our analysis of the method of this section uses the properties of a direction computed by solving the following system of equations:

$$\begin{bmatrix} M & -I \\ Y & X \end{bmatrix} \begin{pmatrix} \bar{d}_x \\ \bar{d}_y \end{pmatrix} = \begin{pmatrix} 0 \\ Xy \end{pmatrix}. \tag{3.3}$$

$(\bar{d}_x, \bar{d}_y)$ is the Newton direction at (x, y) for the system:

$$\begin{aligned} Mx - y &= q, \\ Xy &= 0. \end{aligned}$$

Let $\bar{D}_x = \operatorname{diag}(\bar{d}_x)$. $(\bar{d}_x, \bar{d}_y)$ computed in this way is used as the predictor direction in Ye and Anstreicher [21] and Ji, Potra, and Huang [2]. This Newton direction satisfies the following equalities and inequalites:

$$D^{-1}\bar{d}_x + D\bar{d}_y = v, \tag{3.4}$$

$$\left\|D^{-1}\bar{d}_x + D\bar{d}_y\right\|^2 = x^T y, \tag{3.5}$$

$$\bar{d}_x{}^T \bar{d}_y \geq 0. \tag{3.6}$$

Proofs for (3.4), (3.5), and (3.6) are straightforward. The next four lemmas are used in the proof of Theorem 3.5, which is the main result of this section. This theorem specifies a value of γ in Theorem 2.6.

LEMMA 3.1. *Let* (d_x, d_y) *be the optimal solution of* (3.1). *Then*

$$x^T y \le x^T d_y + y^T d_x. \tag{3.7}$$

Furthermore, for any solution (d_x, d_y) *that is feasible for* (3.1), *we have*

$$x^T d_y + y^T d_x \le \sqrt{2} x^T y. \tag{3.8}$$

Proof. We prove (3.7) by constructing a feasible solution of (3.1) with the objective value $x^T y$. We use the Newton direction for this purpose. For the Newton direction we have

$$\left\| \begin{matrix} D^{-1}\bar{d}_x \\ D\bar{d}_y \end{matrix} \right\|^2 = \left\| D^{-1}\bar{d}_x + D\bar{d}_y \right\|^2 - 2\bar{d}_x{}^T \bar{d}_y \le \left\| D^{-1}\bar{d}_x + D\bar{d}_y \right\|^2 = x^T y.$$

Therefore, $(\bar{d}_x, \bar{d}_y)$ is feasible for (3.1). The objective value of (3.1) at $(\bar{d}_x, \bar{d}_y)$ is

$$y^T \bar{d}_x + x^T \bar{d}_y = x^T y.$$

Hence, the optimal solution of (3.1) must satisfy $x^T d_y + y^T d_x \ge x^T y$. To prove the second statement, it is sufficient to show that (3.8) holds for the optimal solution of (3.1). Now

$$x^T d_y + y^T d_x = w^T \begin{pmatrix} g_x \\ g_y \end{pmatrix} = \rho w^T P_B w = \sqrt{x^T y}\, \|P_B w\| \le \sqrt{2}\, x^T y,$$

because $\|P_B w\| \le \|w\| = \sqrt{2}\, x^T y$. □

LEMMA 3.2. *Let* $(x, y) \in \mathcal{N}_2(\beta)$ *and* (d_x, d_y) *be any feasible solution of* (3.1). *Let* $D_x = \operatorname{diag}(d_x)$. *Then*

(a) $d_x^T d_y = g_x^T g_y \le x^T y/2$,

(b) $\|X d_y + Y d_x\| \le x^T y \sqrt{2(1+\beta)}/\sqrt{n}$,

(c) $\|D_x d_y\| \le x^T y/2$.

Furthermore, if (d_x, d_y) *is an optimal solution of* (3.1) *then, for* $\theta \ge 0$, *we have*

(d) $(x - \theta d_x)^T (y - \theta d_y) \le (1 - \theta + \theta^2/2) x^T y$.

Proof. (a) The first inequality below uses the Cauchy–Schwarz inequality. The second inequality follows from (3.2). Now

$$d_x^T d_y = g_x^T g_y \le \|g_x\|\, \|g_y\| = \frac{\|g_x\|^2 + \|g_y\|^2}{2} \le x^T y/2.$$

(b) First, from Lemma 3.2(a), we have

$$\|g_x + g_y\|^2 = g_x^T g_x + g_y^T g_y + 2 g_x^T g_y \le x^T y + x^T y = 2 x^T y.$$

Now

$$\begin{aligned} \|X d_y + Y d_x\|^2 &= \left\| X^{1/2} Y^{1/2} (g_x + g_y) \right\|^2 \\ &\le \max_i (x_i y_i)\, \|g_x + g_y\|^2 \\ &\le \frac{2(1+\beta)(x^T y)^2}{n}, \end{aligned}$$

where the last inequality follows by using (2.3).

(c) The last inequality below uses the proof of Lemma 3.2(a). Clearly,

$$\|D_x d_y\| = \left\|(D^{-1}D_x)(Dd_y)\right\| \le \left\|D^{-1}d_x\right\| \|Dd_y\| = \|g_x\| \|g_y\| \le x^T y/2.$$

(d) The inequality below follows from Lemma 3.1 and Lemma 3.2(a).

$$\begin{aligned}(x-\theta d_x)^T(y-\theta d_y) &= x^T y - \theta(x^T d_y + y^T d_x) + \theta^2 d_x^T d_y \\ &\le x^T y - \theta x^T y + \tfrac{1}{2}\theta^2 x^T y \\ &= (1-\theta+\theta^2/2)x^T y. \qquad \square\end{aligned}$$

The next lemma is technical. It is used in the proof of Lemma 3.4.

LEMMA 3.3. *Let $(x,y) \in \mathcal{N}_2(\beta)$ and (d_x,d_y) be any feasible solution of* (3.1). *Then*

$$\left\|Xd_y + Yd_x - \frac{1}{n}(x^T d_y + y^T d_x)e\right\| \le \frac{(2+2\beta)^{1/2}x^T y}{\sqrt{n}}$$

and

$$\left\|D_x d_y - \frac{d_x^T d_y}{n}e\right\| \le x^T y/2.$$

Proof. The last inequality below uses Lemma 3.2(b).

$$\begin{aligned}\left\|Xd_y + Yd_x - \frac{1}{n}(x^T d_y + y^T d_x)e\right\|^2 &= \|Xd_y + Yd_x\|^2 - \frac{1}{n}(x^T d_y + y^T d_x)^2 \\ &\le \|Xd_y + Yd_x\|^2 \\ &\le \frac{2(1+\beta)(x^T y)^2}{n}.\end{aligned}$$

Now we prove the second inequality. The first inequality below uses Lemma 3.2(c). Now

$$\left\|D_x d_y - \frac{d_x^T d_y}{n}e\right\|^2 = \|D_x d_y\|^2 - \frac{1}{n}(d_x^T d_y)^2 \le \|D_x d_y\|^2 \le (x^T y)^2/4. \qquad \square$$

LEMMA 3.4. *Let $(x,y) \in \mathcal{N}_2(\beta)$, $0 \le \beta \le 1/6$, and (d_x,d_y) be any feasible solution of* (3.1). *Let $x(\theta) = x - \theta d_x$ and $y(\theta) = y - \theta d_y$. Let $\bar{\theta}$ be the largest θ such that*

$$\beta + \theta(2+2\beta)^{1/2}\sqrt{n} + \theta^2 n/2 \le 2(1-\theta\sqrt{2})/6. \tag{3.9}$$

Then $(x(\theta),y(\theta)) \in \mathcal{N}_2(2/6)$ for $0 \le \theta \le \bar{\theta}$. Furthermore, $\bar{\theta} \ge 0.073/\sqrt{n}$.

Proof. Let $D_x = \operatorname{diag}(d_x)$. The last inequality below uses Lemma 3.3.

$$\begin{aligned}\|X(\theta)y(\theta) - \mu(\theta)e\| &= \left\|(X-\theta D_x)(y-\theta d_y) - \frac{(x-\theta d_x)^T(y-\theta d_y)}{n}e\right\| \\ &= \left\|(Xy-\mu e) - \theta\left(Xd_y + Yd_x - \frac{1}{n}(x^T d_y + y^T d_x)e\right)\right. \\ &\qquad \left. + \theta^2\left(D_x d_y - \frac{d_x^T d_y}{n}e\right)\right\|\end{aligned}$$

$$\begin{aligned}
&\leq \|Xy - \mu e\| + \theta \left\| Xd_y + Yd_x - \frac{1}{n}(x^T d_y + y^T d_x)e \right\| \\
&\quad + \theta^2 \left\| D_x d_y - \frac{d_x^T d_y}{n} e \right\| \\
&\leq \beta\mu + \theta(2 + 2\beta)^{1/2}\sqrt{n}\mu + \theta^2 n\mu/2.
\end{aligned}$$

Therefore, to ensure that $(x(\theta), y(\theta)) \in \mathcal{N}_2(2/6)$, we need to compute a θ such that

$$\left(\beta + \theta(2 + 2\beta)^{1/2}\sqrt{n} + \theta^2 n/2\right)\mu \leq 2\mu(\theta)/6.$$

The second inequality below uses Lemma 3.1. Since

$$\begin{aligned}
x(\theta)^T y(\theta) &= x^T y - \theta(x^T d_y + y^T d_x) + \theta^2 d_x^T d_y \\
&\geq x^T y - \theta(x^T d_y + y^T d_x) \geq x^T y - \theta\sqrt{2}(x^T y) = (1 - \theta\sqrt{2})x^T y,
\end{aligned}$$

it is sufficient to find a θ satisfying (3.9). We can now verify that for $0 \leq \beta \leq 1/6$ and $\theta \leq 0.073/\sqrt{n}$, (3.9) holds. □

THEOREM 3.5. *Let* $(x, y) \in \mathcal{N}_2(\beta)$, $0 \leq \beta \leq 1/6$, *and* (d_x, d_y) *be the optimal solution of* (3.1). *Let* $\theta = 0.073/\sqrt{n}$, *and* $\hat{x} = x - \theta d_x$, $\hat{y} = y - \theta d_y$. *Let* (p_x, p_y) *be the solution of* (2.2) *and* $x^+ = \hat{x} - p_x$, $y^+ = \hat{y} - p_y$. *Then*

$$\frac{x^{+^T} y^+}{x^T y} \leq 1 - \frac{0.013}{\sqrt{n}}$$

and $(x^+, y^+) \in \mathcal{N}_2(1/6)$.

Proof. From Lemma 2.3 and Lemma 3.2(d) we have

$$\begin{aligned}
x^{+^T} y^+ &\leq \left(1 + \frac{2\beta^2}{(1 - 2\beta)n}\right)\hat{x}^T\hat{y} \\
&\leq \left(1 + \frac{2\beta^2}{(1 - 2\beta)n}\right)(1 - \theta + \theta^2/2)x^T y \\
&\leq \left(1 + \frac{1}{12n}\right)\left(1 - \frac{0.073}{\sqrt{n}} + \frac{(0.073)^2}{2n}\right)x^T y.
\end{aligned}$$

One can verify that for $n \geq 2$,

$$\left(1 + \frac{1}{12n}\right)\left(1 - \frac{0.073}{\sqrt{n}} + \frac{(0.073)^2}{2n}\right) \leq 1 - \frac{0.013}{\sqrt{n}}.$$

Lemma 3.4 ensures that $(x^+, y^+) \in \mathcal{N}_2(1/6)$. □

Theorem 3.5 ensures that if we start with a solution $(x^0, y^0) \in \mathcal{N}_2(\beta)$, the iterates (x^k, y^k) generated by the predictor-corrector method also satisfy $(x^k, y^k) \in \mathcal{N}_2(\beta)$. Hence, using Theorem 2.6, we have an $O(\sqrt{n}L)$ iteration method.

3.3. Trust Region Method I. In §3.1, we considered minimizing a linear objective over an ellipsoidal approximation of the feasible region for computing the predictor step. The LCP can be solved by solving the following quadratic program:

$$\begin{array}{ll}
\min & f(x, y) = x^T y, \\
\text{subject to} & Mx - y = q, \\
& x \geq 0,\ y \geq 0.
\end{array} \tag{3.10}$$

The linear objective that is used in the subproblem considered in Ball Optimization Method I can be viewed as a linear approximation of the objective function $f(x, y)$. It is also possible to develop a method for the LCP that optimizes the quadratic objective over ellipsoidal approximations of the feasible region to compute the predictor step. The subproblem solved at each step is now a trust-region problem. The trust-region problem is well studied (see Dennis and Schnabel [1]). A method for solving this problem was recently analyzed by Ye [17]. The reason for considering solutions of trust-region problems is the expectation that by solving a problem representing a better approximation, we might reduce the total number of iterations required to solve the problem. The reduction in the number of iteration should offset the cost of solving a more difficult problem. We now give a method analogous to Ball Optimization Method I.

In this method we compute the predictor direction by solving

$$\begin{array}{ll} \min & (x - d_x)^T (y - d_y), \\ \text{subject to} & Md_x - d_y = 0, \\ & \left\| \begin{array}{c} D^{-1} d_x \\ D d_y \end{array} \right\|^2 \leq \alpha^2 x^T y. \end{array} \tag{3.11}$$

Upon substituting $d_y = Md_x$, the above problem reduces to

$$\begin{array}{rl} \min & x^T y + d_x^T M d_x - (y + M^T x)^T d_x, \\ \text{subject to} & d_x^T [M^T D^2 M + D^{-2}] d_x \leq \alpha^2 x^T y. \end{array}$$

This is a trust-region problem that minimizes a convex quadratic function over an ellipsoidal constraint.

We state the following theorem for a method that uses the predictor direction as a solution of the subproblem defined in this subsection. The proof of this theorem is similar to the proof of Theorem 3.5. We only need to observe that the bound in Lemma 3.2(d) applies for the objective value of (3.11).

THEOREM 3.6. *Let* $(x, y) \in \mathcal{N}_2(\beta)$, $0 \leq \beta \leq 1/6$, *and* (d_x, d_y) *be the optimal solution of* (3.11). *Let* $\hat{x} = x - d_x$, $\hat{y} = y - d_y$, $x^+ = \hat{x} - p_x$, *and* $y^+ = \hat{y} - p_y$. *Then, for* $\alpha = 0.073/\sqrt{n}$,

$$\frac{x^{+^T} y^+}{x^T y} \leq 1 - \frac{0.013}{\sqrt{n}}$$

and $(x^+, y^+) \in \mathcal{N}_2(1/6)$.

4. Methods using affine scaling ellipsoids. The methods described in this section compute the predictor direction at each iteration by solving subproblems that use ellipsoids defined by an affine scaling of the feasible region. Kojima, Megiddo, and Ye [5] used a related (but not the same) problem for computing search directions in the context of a potential reduction algorithm. They proved an $O(n^2 L)$ iteration bound for their algorithm. The algorithms of this section require $O(\sqrt{n} L)$ iterations.

The first method, Ball Optimization Method II, computes the predictor direction by solving the following subproblem:

$$\begin{array}{ll} \text{maximize} & d_x^T y + d_y^T x, \\ \text{subject to} & Md_x - d_y = 0, \\ & \left\| \begin{array}{c} X^{-1} d_x \\ Y^{-1} d_y \end{array} \right\|^2 \leq \dfrac{n}{1 - \beta}. \end{array} \tag{4.1}$$

The next method is analogous to Trust Region Method I, except that it uses the ellipsoidal constraint from Ball Optimization Method II to define the trust-region problem. In this method the predictor direction is computed by solving

$$\begin{array}{ll} \min & (x-d_x)^T(y-d_y), \\ \text{subject to} & Md_x - d_y = 0, \\ & \left\| \begin{matrix} X^{-1}d_x \\ Y^{-1}d_y \end{matrix} \right\|^2 \leq \alpha^2 \dfrac{n}{1-\beta}. \end{array} \tag{4.2}$$

4.1. Analysis of methods using affine scaling ellipsoids. We observe that the ellipsoidal approximation in (4.1) and (3.1) are related. This is particularly true near the central path. This observation is used to simplify the analysis for the methods in this section. Let

$$E_1(\nu) \equiv \left\{ (d_x, d_y) \mid \left\| \begin{matrix} D^{-1}d_x \\ Dd_y \end{matrix} \right\|^2 \leq \nu \right\}$$

and

$$E_2(\nu) \equiv \left\{ (d_x, d_y) \mid \left\| \begin{matrix} X^{-1}d_x \\ Y^{-1}d_y \end{matrix} \right\|^2 \leq \nu \right\}.$$

Let $(x, y) \in \mathcal{N}_2(\beta)$; then it is easy to see that

$$(d_x, d_y) \in E_1(x^T y) \Rightarrow (d_x, d_y) \in E_2(n/(1-\beta)),$$

i.e., $E_1(x^T y) \subseteq E_2(n/(1-\beta))$, and

$$(d_x, d_y) \in E_2(n/(1+\beta)) \Rightarrow (d_x, d_y) \in E_1(x^T y),$$

i.e., $E_2(n/(1+\beta)) \subseteq E_1(x^T y)$. As a consequence of this observation, we can prove results analogous to Lemmas 3.1–3.3 that apply for the methods in this section. We omit these proofs here.

Theorems 4.1 and 4.2 give the rate of convergence of the predictor-corrector algorithms for the LCP, which uses predictor directions of Ball Optimization Method II and Trust Region Method II, respectively.

THEOREM 4.1. *Let* $(x, y) \in \mathcal{N}_2(\beta)$, $0 \leq \beta \leq 1/6$, *and* (d_x, d_y) *be the optimal solution of* (4.1). *Let* $\hat{x} = x - \theta d_x$ *and* $\hat{y} = y - \theta d_y$. *Let* (p_x, p_y) *be the solution of* (2.2) *and* $x^+ = \hat{x} - p_x$, $y^+ = \hat{y} - p_y$. *Then, for* $\theta = 0.06185/\sqrt{n}$,

$$\frac{x^{+^T} y^+}{x^T y} \leq 1 - \frac{0.0015}{\sqrt{n}}$$

and $(x^+, y^+) \in \mathcal{N}_2(\beta)$.

THEOREM 4.2. *Let* $(x, y) \in \mathcal{N}_2(\beta)$, $0 \leq \beta \leq 1/6$, *and* (d_x, d_y) *be the optimal solution of* (4.2). *Let* $\hat{x} = x - d_x$, $\hat{y} = y - d_y$, $x^+ = \hat{x} - p_x$, *and* $y^+ = \hat{y} - p_y$. *Then, for* $\alpha = 0.06185/\sqrt{n}$,

$$\frac{x^{+^T} y^+}{x^T y} \leq 1 - \frac{0.0015}{\sqrt{n}},$$

and $(x^+, y^+) \in \mathcal{N}_2(\beta)$.

5. Concluding remarks. We have shown that the generic predictor-corrector framework leads to a variety of interior point methods for solving the linear complementarity problem. All of these methods have the same iteration complexity. The predictor direction used by these methods distinguishes them.

It is important to perform computational tests to find which of these methods is most effective. We have performed some preliminary tests on randomly generated problems. These results indicate that Ball Optimization Method I takes considerably fewer iterations than Ball Optimization Method II and the predictor-corrector method using the Newton direction for problems not satisfying strict complementarity assumption. The number of iterations required by this method appears similar to the number of iterations taken by Trust Region Method I. A detailed discussion of these results is the topic for a subsequent paper.

Acknowledgments. We thank the two referees and the Associate Editor for their careful reading of the paper and the suggestions they made toward improving it. We also thank Professor Michael J. Todd at Cornell University for his helpful comments.

REFERENCES

[1] J. E. DENNIS, JR. AND ROBERT B. SCHNABEL, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1983.

[2] J. JI, F. POTRA, AND S. HUANG. *A predictor-corrector method for linear complementarity problems with polynomial complexity and superlinear convergence*, Report no. 18, University of Iowa, Iowa City, IA, 1991.

[3] S. KAPOOR AND P. VAIDYA, *Fast algorithms for convex quadratic programming and multicommodity flows*, in Proceedings of the 18th Annual Symposium on the Theory of Computing, 1986, pp. 147–159.

[4] N. KARMARKAR, *A new polynomial-time algorithm for linear programming*, Combinatorica, 4 (1984), pp. 373–395.

[5] M. KOJIMA, N. MEGIDDO, AND Y. YE, *An interior point potential reduction algorithm for the linear complementarity problem*, Math. Programming, 54 (1992), pp. 267–279.

[6] M. KOJIMA, S. MIZUNO, AND A. YOSHISE, *An $O(\sqrt{n})L$ iteration potential reduction algorithm for linear complementarity problems*, Math. Programming, 50 (1991), pp. 331–342.

[7] ———, *A polynomial-time algorithm for a class of linear complementarity problems*, Math. Programming, 44 (1989), pp. 1–26.

[8] M. KOJIMA, S. MEGIDDO, T. NOMA, AND A. YOSHISE, *A unified approach to interior point algorithms for linear complementarity problems*, in Lecture Notes in Comput. Sci. 538, Springer-Verlag, New York, 1991.

[9] K. MCSHANE, *On Long Step $O(n^3L)$ Primal-Dual Potential Reduction Algorithms for Linear Programming and The Linear Complementarity Problem*, 1992, manuscript.

[10] S. MEHROTRA, *Quadratic convergence in a primal-dual method*, Math. Oper. Res., 18 (1993), pp. 741–751.

[11] S. MIZUNO, M. TODD, AND Y. YE, *On adaptive-step primal-dual interior-point algorithms for linear programming*, Math. Oper. Res., 18 (1993), pp. 964–981.

[12] R. C. MONTEIRO AND I. ADLER, *Interior path following primal-dual algorithms. Part* I: *Linear programming*, Math. Programming, 44 (1989), pp. 27–41.

[13] ———, *Interior path following primal-dual algorithms. Part* II: *Convex quadratic programming*, Math. Programming, 44 (1989), pp. 43–66.

[14] M. J. TODD. *Projected scaled steepest descent in Kojima-Mizuno-Yoshise's potential reduction algorithm for the linear complementarity problem*, Technical report no. 950, School Of Operations Research and Industrial Engineering, College of Engineering, Cornell University, Ithaca, NY, 1990.

[15] M. J. TODD AND Y. YE, *A centered projective algorithm for linear programming*, Math. Oper. Res., 15 (1990), pp. 508–529.

[16] Y. YE, *A further result on the potential reduction algorithm for the P-matrix linear complementarity problem*, Technical report, Department of Management Sciences, University of Iowa, Iowa City, IA.

[17] ———, *A new complexity result on minimization of a quadratic function with a sphere constraint*, Technical report, Department of Management Sciences, University of Iowa, Iowa City, IA, 1990.

[18] Y. YE, O. GÜLER, R. A. TAPIA, AND Y. ZHANG, *A quadratically convergent $O(\sqrt{n}L)$-iteration algorithm for linear programming*, Math. Programming, 59 (1993), pp. 151–162.

[19] Y. YE AND P. PARDALOS, *A class of linear complementarity problems solvable in polynomial time*, Linear Algebra Appl., 152 (1991), pp. 3–17.

[20] Y. YE AND E. TSE, *An extension of Karmarkar's projective algorithm for convex quadratic programming*, Math. Programming, 44 (1989), pp. 157–179.

[21] Y. YE AND K. ANSTREICHER, *On quadratic and $O(\sqrt{n}L)$ convergence of a predictor-corrector algorithm for* LCP, Math. Programming, 62 (1993), pp. 537–551.

SIAM J. OPTIMIZATION
Vol. 4, No. 3, pp. 455–460, August 1994
© 1994 Society for Industrial and Applied Mathematics

THE MONOTONIC DIAMETER OF THE PERFECT 2-MATCHING POLYTOPE*

FRED J. RISPOLI†

Abstract. The monotonic diameter of the perfect 2-matching polytope associated to the complete graph with n nodes is shown to be $[n/2]$ for all $n \geq 3$, except for $n = 3, 4, 8$, and 9, in which case the monotonic diameter is $[n/2] - 1$. The result establishes the monotonic Hirsch conjecture for this polytope.

Key words. polytope, diameter, Hirsch conjecture

AMS subject classification. 90C35

1. Introduction. The *distance* between any pair of extreme points of a polytope P is the number of edges in a shortest path linking them. The *diameter* of a polytope P, $\delta(P)$, is the greatest distance between any pair of extreme points of P. Many results are known concerning the function δ, most of which are contained in the comprehensive survey paper by Klee and Kleinschmidt [5]. Now let ϕ be a linear form defined on P and $\mathbf{x}$ any extreme point. Denote by $\delta(P, \phi, \mathbf{x})$ the number of edges of P in a shortest path of pairwise neighboring extreme points $\mathbf{x} = \mathbf{x}^0, \mathbf{x}^1, \ldots, \mathbf{x}^t = \mathbf{x}^*$ linking $\mathbf{x}$ to some extreme point $\mathbf{x}^*$ minimizing ϕ, such that $\phi(\mathbf{x}^s) \geq \phi(\mathbf{x}^{s+1})$. Let

$$\delta(P, \phi) = \max\{\delta(P, \phi, \mathbf{x}) : \mathbf{x} \text{ is an extreme point of } P\}.$$

Then the *monotonic diameter* of P is

$$\delta^*(P) = \max\{\delta(P, \phi) : \phi \text{ is a linear form defined on } P\}.$$

Observe that for any polytope P, $\delta^*(P)$ exists and $\delta(P) \leq \delta^*(P)$.

The function δ^* is of interest in linear programming because it is related to the best possible behavior of edge following algorithms requiring monotonicity of the objective function such as the simplex method. The function δ^* has been determined for various polytopes including the perfect matching and shortest path polytopes (Rispoli [8]) and for a class of transportation polytopes (Balinski and Rispoli [1]). These results also show that $\delta(P)$ may be different from $\delta^*(P)$. For example, Padberg and Rao [7] showed that the perfect matching polytope associated to the complete graph K_n has a diameter of two, independent of n. However, the monotonic diameter of this polytope is $[n/4]$. In this note it is shown that the monotonic diameter of the perfect 2-matching polytope associated to K_n is $[n/2]$, for all $n \geq 3$, except for $n = 3, 4, 8$, and 9, in which case it is $[n/2] - 1$. The proof is by induction and requires four specific cases to anchor the induction. This note concludes with a brief discussion concerning the monotonic Hirsch conjecture.

2. The perfect 2-matching polytope. Let K_n be the complete graph on nodes $N = \{1, 2, \ldots, n\}$ with edges $E = \{\{i, j\} : i \in N, j \in N, i \neq j\}$. Given any subgraph H of K_n, associate the vector $\mathbf{x}^H \in \mathbf{R}^E$ called the *incidence vector* of H defined by $\mathbf{x}_e^H = 1$, if edge $e \in H$, otherwise, $\mathbf{x}_e^H = 0$. An edge set M is called a *perfect 2-matching* if M is a subgraph of K_n having n edges and all nodes of degree 2. The *perfect 2-matching polytope* is

$$Q_{2M}^n = \text{conv}\{\mathbf{x}^M : M \text{ is a perfect 2-matching in } K_n\}.$$

The *perfect 2-matching problem* is minimize$\{\phi(\mathbf{x}) = \mathbf{cx} : \mathbf{x} \in Q_{2M}^n\}$, where $\mathbf{c} = (c_{ij})$ are arbitrary reals. Note that every Hamiltonian cycle in K_n is a perfect 2-matching. Thus, the

* Received by the editors November 16, 1992; accepted for publication (in revised form) June 4, 1993.
† Department of Mathematics, Dowling College, Oakdale, New York 11769.

perfect 2-matching problem is a relaxation of the traveling salesman problem. Moreover, a complete and nonredundant system of equations and inequalities describing Q^n_{2M} is known, making the perfect 2-matching polytope helpful in determining facets of the traveling salesman polytope (see [3] for details).

For any two sets S and T, let $S \Delta T$ denote the *symmetric difference* $(S \sim T) \cup (T \sim S)$. Let M_1 and M_2 be any pair of perfect 2-matchings in K_n; a cycle in $M_1 \Delta M_2$ is called *alternating* if its edges alternate between M_1 and M_2 edges. Note that if $M_1 \Delta M_2$ is connected, then this set forms an alternating cycle itself. It is well known that the extreme points of Q^n_{2M} are in 1–1 correspondence with the perfect 2-matchings of K_n.

LEMMA 1. *Let M_1 and M_2 be any pair of perfect 2-matchings in K_n. Then the extreme points of Q^n_{2M} corresponding to M_1 and M_2 are neighbors if and only if $M_1 \Delta M_2$ contains a unique alternating cycle.*[1]

Proof. Suppose the extreme points $\mathbf{x}^{M_1}$ and $\mathbf{x}^{M_2}$ are neighbors. Then $M_1 \Delta M_2$ must have at least one alternating cycle. For $M_1 \neq M_2$ implies that there is an edge $e_1 \in M_1 \sim M_2$, say, $e_1 = \{i_1, i_2\}$. Since i_2 has degree 2 in M_2, there is an edge $e_2 = \{i_2, i_3\} \in M_2 \sim M_1$. Repeating this observation we trace out a path that must eventually return to i_1, forming a cycle, say, C. If C is not an alternating cycle, then C is an odd cycle with both edges incident to i_1 contained in $M_1 \sim M_2$. But then there must be an edge $f_1 = \{i_1, j_1\} \in M_2 \sim M_1$. Proceeding as before, trace out another path that must return to i_1 via an edge in $M_2 \sim M_1$. This creates an alternating cycle. If there are two or more alternating cycles in $M_1 \Delta M_2$, it is possible to construct perfect 2-matchings H_1 and H_2, distinct from M_1 and M_2, such that $H_1 \cup H_2 = M_1 \cup M_2$. Let C_1 and C_2 be disjoint alternating cycles in $M_1 \Delta M_2$. Define perfect 2-matchings H_1 and H_2 as follows. Set

$$H_1 = [M_1 \sim (C_1 \cap M_1)] \cup (C_1 \cap M_2)$$

and

$$H_2 = [M_2 \sim (C_1 \cap M_2)] \cup (C_1 \cap M_1).$$

Then, $H_1 \neq M_1$ since H_1 contains an edge in $M_2 \sim M_1$. $H_1 \neq M_2$ since there is an edge in $C_2 \cap M_2$ that is not in H_1. Similarly, $H_2 \neq M_1$ and $H_2 \neq M_2$. But now

$$\frac{1}{2}\mathbf{x}^{M_1} + \frac{1}{2}\mathbf{x}^{M_2} = \frac{1}{2}\mathbf{x}^{H_1} + \frac{1}{2}\mathbf{x}^{H_2},$$

contradicting that $\mathbf{x}^{M_1}$ and $\mathbf{x}^{M_2}$ are neighbors.

Showing that $M_1 \Delta M_2$ contains a unique alternating cycle implies that M_1 and M_2 are neighbors is straightforward, and the details are omitted. □

For any real number α, let $[\alpha]$ denote the largest integer less than or equal to α.

LEMMA 2. *It holds that $\delta^*(Q^n_{2M}) \leq [n/2]$ for all $n \geq 3$.*

Proof. Let $\phi(\mathbf{x}) = \mathbf{cx}$ be any linear form defined on Q^n_{2M} and $\mathbf{x}^*$ any extreme point on which ϕ attains its minimum. Let $\mathbf{x}$ be any other extreme point, and let M and M^* be the perfect 2-matchings corresponding to $\mathbf{x}$ and $\mathbf{x}^*$, respectively. It is shown that an extreme point $\mathbf{y}$ can always be obtained satisfying: (i) $\mathbf{x}$ and $\mathbf{y}$ are neighbors; (ii) if M_1 is the perfect 2-matching corresponding to $\mathbf{y}$, then M_1 contains at least two more edges of M^* than M; (iii) $\phi(\mathbf{x}) \geq \phi(\mathbf{y})$.

The first part of the proof of Lemma 1 shows that $M \neq M^*$ implies that there is at least one alternating cycle in $M \Delta M^*$. If $M \Delta M^*$ contains a unique alternating cycle, then $\mathbf{x}$

[1] The cycle may visit a node more than once, but edges are traversed at most once.

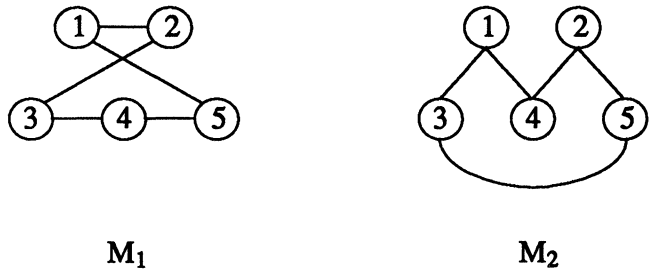


FIG. 1.

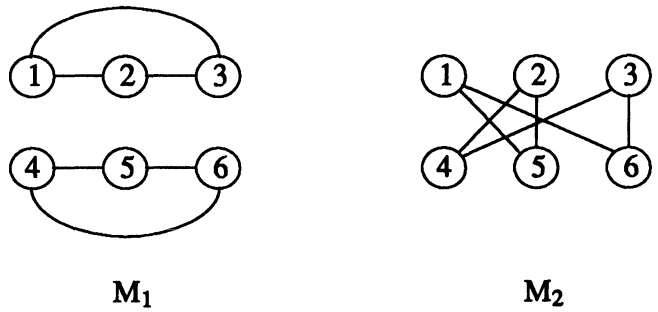


FIG. 2.

and $\mathbf{x}^*$ are neighbors so take $\mathbf{y} = \mathbf{x}^*$. Otherwise, there are at least two alternating cycles in $M \Delta M^*$. Let C be an alternating cycle that is inclusion-wise minimal; that is, C does not contain an alternating cycle as a proper subset. Define $\mathbf{y}$ to be the extreme point corresponding to the perfect 2-matching

$$M_1 = \{M \cup (M^* \cap C)\} \sim (M \cap C).$$

Then $M_1 \Delta M = C$, hence $\mathbf{x}$ and $\mathbf{y}$ are neighbors. M_1 must contain at least two more edges of M^* since C must be an even cycle. Finally, $\phi(\mathbf{x}) \geq \phi(\mathbf{y})$ for otherwise the edges in $M \cap C$ weigh less than the edges in $M^* \cap C$. Consequently,

$$\{M^* \cup (M \cap C)\} \sim (M^* \cap C)$$

is a perfect 2-matching corresponding to, say, $\mathbf{z}$, and $\phi(\mathbf{z}) < \phi(\mathbf{x}^*)$ a contradiction. □

LEMMA 3. *It holds that* (a) $\delta^*(Q^5_{2M}) = 2$, (b) $\delta^*(Q^6_{2M}) = 3$, (c) $\delta^*(Q^7_{2M}) = 3$, *and* (d) $\delta^*(Q^{10}_{2M}) = 5$.

(a) By Lemma 2, $\delta^*(Q^5_{2M}) \leq 2$. Define perfect 2-matchings M_1 and M_2 given in Fig. 1. Since $M_1 \Delta M_2$ has more than one alternating cycle, the extreme points corresponding to M_1 and M_2 are not neighbors. Therefore, $\delta^*(Q^5_{2M}) \geq \delta(Q^5_{2M}) \geq 2$.

(b) By Lemma 2, $\delta^*(Q^6_{2M}) \leq 3$. Define perfect 2-matchings M_1 and M_2 given in Fig. 2. It takes at least three steps to go from M_1 to M_2. At most two edges of M_2 can be obtained at the first step, leaving four edges to be subsequently obtained. However, at most three edges of M_2 may be obtained in any one step, since any inclusion-wise minimal alternating cycle on at most six nodes has at most six edges. Thus, $\delta^*(Q^6_{2M}) \geq \delta(Q^6_{2M}) \geq 3$.

(c) By Lemma 2, $\delta^*(Q^7_{2M}) \leq 3$. Define perfect 2-matchings given in Fig. 3. The extreme points corresponding to M_1 and M_2 are not adjacent since $M_1 \Delta M_2$ contains more than one

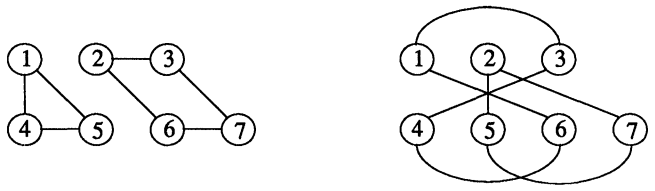


FIG. 3.

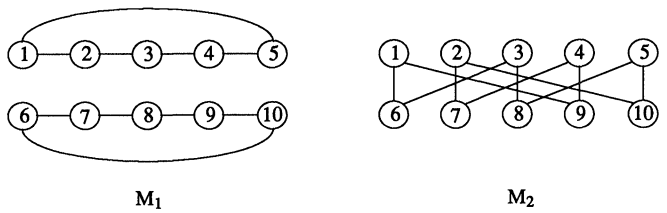


FIG. 4.

alternating cycle. Moreover, there is no intermediate extreme point $\mathbf{x}^M$ adjacent to both $\mathbf{x}^{M_1}$ and $\mathbf{x}^{M_2}$ since the existence of such a perfect 2-matching M would imply the existence of an alternating cycle in $M_1 \Delta M_2$ of eight edges of the form a, b, c, a, d, e, f, g, a. No such cycle exists.

(d) Again Lemma 2 gives $\delta^*(Q^{10}_{2M}) \leq 5$. Define perfect 2-matchings M_1 and M_2 given in Fig. 4. Observe that there are no alternating cycles in $M_1 \Delta M_2$ with six edges. Moreover, from a careful examination of possible alternating cycles with eight or ten edges, it follows that in all cases the net gain is not more that two. Hence at least five steps are needed. □

THEOREM 1. *It holds that* $\delta^*(Q^n_{2M}) = [n/2]$ *for all* $n \geq 3$, *except for* $n = 3, 4, 8,$ *and* 9, *in which case* $\delta^*(Q^n_{2M}) = [n/2] - 1$.

Proof. Clearly, $\delta^*(Q^3_{2M}) = 0$ and $\delta^*(Q^4_{2M}) = 1$. In addition, tedious proofs exist showing that $\delta^*(Q^8_{2M}) = 3 = \delta^*(Q^9_{2M})$; details are omitted here. Also note that $\delta^*(Q^{14}_{2M}) = 7$. This follows from Lemma 2 and the fact that the extreme points given in Fig. 5 require seven steps to be linked. The argument is similar to the proof of Lemma 3(d).

The remainder of the proof is by induction on n and requires the four special cases treated in Lemma 3 to anchor the induction. Consider any $n \geq 11$, $n \neq 14$. By Lemma 2, $\delta^*(Q^n_{2M}) \leq [n/2]$. By the inductive hypothesis employed on either Q^{n-6}_{2M} or Q^{n-10}_{2M}, whichever is appropriate, there is a linear form $\phi(\mathbf{x}) = \mathbf{cx}$ and an extreme point $\mathbf{x}$ requiring either $[(n-6)/2]$ or $[(n-10)/2]$ steps to be linked to an extreme point $\mathbf{x}^*$ minimizing ϕ by a path nonincreasing in ϕ. Suppose, for convenience, that the inductive hypothesis is employed on Q^{n-6}_{2M} and $[(n-6)/2]$ steps are required to link $\mathbf{x}^M$ to $\mathbf{x}^{M^*}$ on Q^{n-6}_{2M}. Define perfect 2-matchings in K_n by $H = M \cup M_1$ and $H^* = M^* \cup M_2$ where M_1 and M_2 are the perfect 2-matchings given in Fig. 2 on nodes $n-5, n-4, \ldots, n$. Define the linear form $\phi'(\mathbf{x}) = \mathbf{c}'\mathbf{x}$ on Q^n_{2M} as follows:

$$
\begin{aligned}
c'_{ij} &= c_{ij} && \text{if } i \leq n-6 \text{ and } j \leq n-6;\\
c'_{ij} &= 1 && \text{if } \{i,j\} \in M_1;\\
c'_{ij} &= 0 && \text{if } \{i,j\} \in M_2;\\
c'_{ij} &> \alpha + 6 && \text{otherwise,}
\end{aligned}
$$

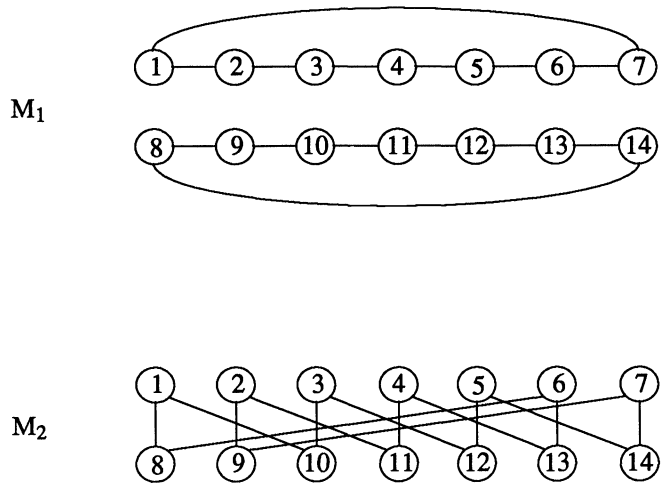


FIG. 5.

where α is prohibitively large, say, greater than the sum of the c_{ij}. Then $\mathbf{x}^{H^*}$ is the only extreme point of Q^n_{2M} minimizing ϕ', and $[(n-6)/2]+3=[n/2]$ steps are required to link $\mathbf{x}^H$ to $\mathbf{x}^{H^*}$ on Q^n_{2M} by a path nonincreasing in ϕ'. □

3. The monotonic Hirsch conjecture. The Hirsch conjecture states that if a convex polyhedron P is defined by q facets, then $\delta(P) \leq q-$ dim P. The monotonic Hirsch conjecture requires that $\delta^*(P) \leq q -$ dim P. Todd [9] has shown that the monotonic Hirsch conjecture is false for general polytopes of dimension 4 or more. Nevertheless, the monotonic Hirsch conjecture is true for some important classes of polytopes including all three-dimensional polytopes (Klee [4]), Leontief substitution polyhedra (Grinold [2]), and the polytopes in [1] and [8]. It is also an easy exercise to use the greedy algorithm to prove the monotonic Hirsch conjecture for matroid polytopes. In the case of the perfect 2-matching polytope, we know that the dimension of Q^n_{2M} is $\binom{n}{2}-n$ and Q^n_{2M} has more than $\binom{n}{2}$ facets; see [3]. Therefore,

$$q - \dim Q^n_{2M} \geq \binom{n}{2} - \left[\binom{n}{2} - n\right] = n > \left[\frac{n}{2}\right] \geq \delta^*(Q^n_{2M})$$

and Lemma 2 proves the monotonic Hirsch conjecture for Q^n_{2M}.

REFERENCES

[1] M. L. BALINSKI AND F. J. RISPOLI, *Signature classes of transportation polytopes*, Math. Programming, 60 (1993), pp. 127–144.

[2] R. C. GRINOLD, *The Hirsch conjecture in Leontief substitution systems*, SIAM J. Appl. Math., 21 (1970), pp. 483–485.

[3] M. GRÖTSCHEL AND M. PADBERG, *Polyhedral theory*, in The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization, E. Lawler, J. K. Lenstra, A. R. Kan, and D. Shmoys, eds., Wiley, New York, 1986.

[4] V. KLEE, *Paths on Polyhedra* I, J. Soc. Indust. Appl. Math. 13 (1965), pp. 946–956.

[5] V. KLEE AND P. KLEINSCHMIDT, *The d-step conjecture and its relatives*, Math. Oper. Res., 12 (1987), pp. 718–755.

[6] E. LAWLER, J. K. LENSTRA, A. R. KAN, AND D. SHMOYS, EDS., *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*, Wiley, New York, 1986.

[7] M. W. Padberg and M. R. Rao, *The travelling salesman problem and a class of polyhedra of diameter two*, Math. Programming, 7 (1974), pp. 32–45.

[8] F. J. Rispoli, *The monotonic diameter of the perfect matching and shortest path polytopes*, Oper. Res. Lett., 12 (1992), pp. 23–27.

[9] M. Todd, *The monotonic bounded Hirsch conjecture is false for dimension at least 4*, Math. Oper. Res., 5 (1980), pp. 599–601.

SIAM J. OPTIMIZATION
Vol. 4, No. 3, pp. 461–485, August 1994
© 1994 Society for Industrial and Applied Mathematics
002

LOCAL CONVERGENCE OF A TWO-PIECE UPDATE OF A PROJECTED HESSIAN MATRIX*

CHAYA GURWITZ†

Abstract. The problem considered is that of minimizing a nonlinear function subject to a set of equality constraints. Recent research has focused on methods that solve such problems by recurring an approximation to second derivative information projected onto the tangent space of the constraints. The motivation for these methods is that only the projected matrix enters into the optimality conditions for the nonlinear problem and that the reduced matrix can be recurred by means of updates that maintain hereditary positive definiteness. However, such methods can achieve at best local two-step Q-superlinear convergence. The algorithm presented here performs a two-piece update of a "one-sided projected Hessian matrix." This method maintains one piece of the projected Hessian as a positive definite matrix and uses Broyden's method for updating the other piece. It is shown that if at least one of the updates is performed at each iteration, the method is locally one-step Q-superlinearly convergent.

Key words. constrained optimization, nonlinear programming, projected Hessian, reduced Hessian, Newton's method, sequential quadratic programming

AMS subject classifications. 90C30, 65K05, 49M37

1. Introduction.

Consider the problem of minimizing a nonlinear function subject to a set of nonlinear equality constraints

$$\begin{aligned} &\min_{x\in\mathbf{R}^n} f(x) \\ &\text{s.t. } c_i(x) = 0 \quad i = 1,\dots,m, \end{aligned} \tag{1.1}$$

where the objective and constraint functions are twice continuously differentiable. We are concerned with developing algorithms to find a local minimum of (1.1) by using approximations to second derivative information in the tangent space to the constraints. The main motivation for work in this area is that only the projected matrix enters into the optimality conditions for the nonlinear problem.

The convergence rate that has been obtained for various reduced Hessian updating algorithms is two-step Q-superlinear. In this paper a method is presented that involves a Broyden, Fletcher, Goldfarb, Shannon (BFGS) or Davidon, Fletcher, Powell (DFP) update of the reduced Hessian and a Broyden update of the Hessian projected onto another subspace. Section 2 introduces notation and discusses some of the sequential quadratic programming methods that have been used to solve (1.1). The new method, a two-piece update of a "one-sided projected Hessian," is then presented. In §3 it is shown that under certain conditions, if at least one of the updates is performed at each iteration, the algorithm achieves local one-step Q-superlinear convergence, and that if no assumptions are made regarding the updates, the algorithm is at least two-step Q-superlinearly convergent. Finally, in §4, numerical results are presented.

2. Projected Hessian updating algorithms.

The following notation is used in reference to problem (1.1): Let $g(x) \equiv \nabla f(x)$, the gradient of the objective function, $c(x) \equiv [c_1(x),\dots,c_m(x)]^T$ and $A(x) \equiv [\nabla c_1(x),\dots,\nabla c_m(x)]$, the matrix of constraint gradients. It is assumed that the functions f and c_i are twice differentiable, that the Hessians of the objective and constraint functions are Lipschitz continuous matrix functions of x, and that $A(x)$ has full column rank.

* Received by the editors February 6, 1990; accepted for publication (in revised form) April 16, 1993. This work was supported in part by National Science Foundation grant CCR-8713893 and in part by a grant from The City University of New York PSC-CUNY Research Award Program.

† Department of Computer and Information Science, Brooklyn College, Brooklyn, New York 11210.

Let $Z(x)$ be an orthogonal matrix whose columns span the null space of $A(x)^T$, and let $Y(x)$ be an orthogonal matrix whose columns form a basis for the range space of $A(x)$. Such matrices can be obtained, for example, by forming the QR factorization that produces

$$A(x) = [Q(x)] \begin{bmatrix} R(x) \\ 0 \end{bmatrix}. \tag{2.1}$$

Recall that m is the number of constraints and let $t = n - m$. The first m columns of $Q(x)$ can be used to define the columns of $Y(x)$, while the columns of $Z(x)$ can be formed from the last t columns of $Q(x)$.

Although $A(x)$ is a continuous function of x, the QR decomposition does not necessarily produce continuously varying Z and Y [9], [6]. Gill et al. [17] show that the QR factorization of the current matrix, $A(x_k)$, can be computed by updating the QR factorization of $A(x_{k-1})$, the matrix at the previous iteration, in such a way as to produce a continuously varying Q. The results presented in §3 require that the QR factorization be computed in this manner, so that both $Z(x)$ and $Y(x)$ are continuous.

The first-order necessary conditions for a point x_* to be a local minimum of (1.1) are the existence of a vector of Lagrange multipliers, λ_*, such that

$$g(x_*) = A(x_*)\lambda_* \tag{2.2a}$$

and

$$c(x_*) = 0. \tag{2.2b}$$

Condition (2.2a) is equivalent to requiring x_* to be a stationary point of the Lagrangian function

$$L(x, \lambda) = f(x) - \sum_{i=1}^{m} \lambda_i c_i(x)$$

when $\lambda = \lambda_*$. Second-order optimality conditions are expressed in terms of the Hessian with respect to x of the Lagrangian function,

$$W(x, \lambda) = \nabla^2 f(x) - \sum_{i=1}^{m} \lambda_i \nabla^2 c_i(x).$$

A necessary condition for optimality is that the projected Hessian $Z(x_*)^T W(x_*, \lambda_*) Z(x_*)$ is positive semidefinite. Strict positive definiteness of the projected Hessian is a second-order sufficient condition. Proofs of these optimality conditions can be found in [11]. In addition to the assumption that A has full rank, it is assumed throughout that the second-order sufficient conditions hold at x_*.

Algorithms based on *sequential quadratic programming* (SQP) have been found particularly effective for solving nonlinear programming problems. Methods of this type approximate the minimizer of (1.1) by solving a sequence of quadratic programming (QP) subproblems. In these subproblems a local model of the nonlinear problem is posed in which the curvature of the Lagrangian function is approximated by a suitable matrix and linear approximations are made to the constraints. The solution of the QP is used to provide a search direction p_k. The algorithm then determines a step length α_k such that $x_k + \alpha_k p_k$ is, in some sense, a sufficiently better approximation to x_*.

The QP subproblem for (1.1) is given by

$$\min_{p_k \in R^n} \frac{1}{2} p_k^T H_k p_k + p_k^T g_k \qquad \text{s.t.} \quad A_k^T p_k = -c_k,$$

where H_k is some approximation to the Hessian of the Lagrangian function. A first-order estimate of the Lagrange multipliers can be found by solving the least squares problem

$$\min_{\lambda} \|A_k\lambda_k - g_k\|_2^2. \tag{2.3}$$

The solution to the QP subproblem can be easily computed by solving two systems of linear equations. The solution p_k^* can be written as

$$p_k^* = Y_k p_y^* + Z_k p_z^*,$$

where Y_k and Z_k are defined as in (2.1). The vectors p_y^* and p_z^* can be found by solving

$$A_k^T Y_k p_y^* = -c_k, \tag{2.4a}$$

$$Z_k^T H_k Z_k p_z^* = -Z_k^T g_k - Z_k^T H_k Y_k p_y^*. \tag{2.4b}$$

Using the QR factorization (2.1), the matrix R_k can be used in place of $A_k^T Y_k$ in (2.4a). The solution is unique when A_k has full column rank and $Z_k^T H_k Z_k$ is positive definite. This method of computing p_* is discussed by Murray and Wright [26].

Murray [24], Biggs [1], Han [21], Tapia [33], Glad [18], Schittkowski [30], and Gill et al. [15], [16] have all proposed quasi-Newton methods for solving (1.1) in which a full Hessian matrix is updated. These methods are locally and Q-superlinearly convergent. Boggs, Tolle, and Wang [2] and Fontecilla, Steihaug, and Tapia [12] present convergence results for algorithms of this kind and establish characterizations of Q-superlinear convergence. These methods involve approximating the Hessian of either the Lagrangian function or, since W may not be positive definite, an augmented Lagrangian function. This augmenting term can lead to very ill-conditioned matrices even near the solution. Powell [29] uses a modified updating formula that preserves positive-definiteness, but which can generate ill-conditioned approximations and does not give Q-superlinear convergence.

Recent research has centered about developing algorithms that maintain an approximation to the reduced Hessian rather than to the full second derivative matrix. The main motivation for work in this area is that only the projected matrix enters into the optimality conditions for the nonlinear problem. One advantage of these methods is that updating projected second derivative information reduces the dimension of the matrix to be recurred. Since the reduced matrix of second derivatives is known to be positive-definite at the solution, one is able to make use of quasi-Newton updates. Moreover, the necessity of introducing the augmenting term that can lead to ill-conditioned matrices is avoided.

The idea of recurring an approximation to the projected Hessian rather than to the full second derivative matrix was first suggested by Murray and Wright [25]. Some particular algorithms and their convergence analysis were given for the equality constrained case in [14], [7], [27], and [5]. Gurwitz [19] and Gurwitz and Overton [20] present algorithms of this nature for the inequality constrained problem. These methods all approximate Z^TWZ using a positive-definite update such as DFP or BFGS and ignore Z^TWY. The convergence rates of these methods are locally two-step Q-superlinear.

Coleman and Fenyes [8] and Fontecilla [13] presented updates of the full Hessian matrix that are designed to maintain positive-definiteness of the null-space approximation. These updates are applied to the full Hessian, or, in one of the updates proposed by Coleman and Fenyes, to the matrix Z^TW. Although the methods of Coleman and Fenyes are appealing, they require solving a cubic equation at each iteration, and it is sometimes difficult to determine whether a particular root would result in an ill-conditioned update and should be rejected. More recently, Tagliaferro [31] proposed a locally one-step Q-superlinear method where Z^TWZ and

Z^TWY were updated separately. However, the Powell, Symmetric, Broyden (PSB) update was used for Z^TWZ and thus the approximations to the reduced Hessian do not necessarily remain positive-definite. Moreover, the update he used to approximate Z^TWY can become very ill conditioned, especially near the solution.

The algorithm presented in this paper is motivated by the ideas of Fenyes and Tagliaferro. The method generates positive-definite approximations to Z^TWZ and at the same time separately maintains a least-change update to Z^TWY. This method has similar theoretical properties to the one proposed by Tagliaferro and is more useful in practice. The updates of the two-sided projected Hessian Z^TWZ are guaranteed to be positive-definite; the update of Z^TWY is safeguarded so that the approximating matrix remains well defined. The updates used are standard quasi-Newton formulae and, thus, are easy to implement.

Since both Z^TWZ and Z^TWY are maintained, the method produces, in essence, an approximation to the "one-sided" projected Hessian Z^TW. This can be particularly useful in the context of inequality-constrained problems. In the inequality-constrained case, as the approximation to the active set is revised, the definition and dimension of the null space can change. Generally, second derivative information in the new space can be derived from a full Hessian approximation, but is harder to obtain when only the projected Hessian is being recurred. If Z^TW is approximated, better estimates can be obtained for the reduced Hessian in the new space.

Let B_k denote the current approximation to Z^TWZ, and D_k denote the current approximation to Z^TWY. The DFP or BFGS update will be used to compute B_{k+1} and the Broyden update will be used to obtain D_{k+1}. These updates are generally defined in terms of two vectors s_k and y_k that represent the change in the variables and the gradient values. These vectors must be defined differently for the Broyden update of D_k, since the second derivative information is being approximated on a different subspace. The notation $\hat{s}_k$ and $\hat{y}_k$ will be used to denote the corresponding values to be used in performing the Broyden update.

The DFP, BFGS, and Broyden updates are defined respectively by (2.5), (2.6), and (2.7):

$$B_{k+1}^{\text{DFP}} = B_k + \frac{(y_k - B_k s_k)y_k^T + y_k(y_k - B_k s_k)^T}{y_k^T s_k} - \frac{(y_k - B_k s_k)^T s_k y_k y_k^T}{(y_k^T s_k)^2}. \tag{2.5}$$

$$B_{k+1}^{\text{BFGS}} = B_k + \frac{y_k y_k^T}{y_k^T s_k} - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k}. \tag{2.6}$$

$$D_{k+1} = D_k + \frac{(\hat{y}_k - D_k \hat{s}_k)\hat{s}_k^T}{\hat{s}_k^T \hat{s}_k}. \tag{2.7}$$

Nocedal and Overton [27] provide a number of possible choices for the definition of the vectors s_k and y_k used in updating B_k. Essentially, s_k can be chosen to be the null space component of the step from one iterate to the next, and y_k can be chosen as the corresponding change in the Lagrangian function, also projected onto the reduced space. For the sake of clarity we focus on one particular choice and define s_k and y_k by

$$s_k = Z_k^T(x_{k+1} - x_k), \tag{2.8a}$$

$$y_k = Z_k^T(g_{k+1} - A_{k+1}\lambda_{k+1} - g_k). \tag{2.8b}$$

Nocedal and Overton point out that the update (2.5) or (2.6) might fail since $\|s_k\|$ can be small compared to $\|y_k\|$ and, indeed, $s_k^T y_k$ may not be positive. In this event, the updates may be undefined or may cause the Hessian approximation to become unbounded. For this reason, Gabay [14] and Womersley and Fletcher [35] propose using the Powell modified update [29]. However, as Nocedal and Overton note, Powell's modified update can still result

in an unbounded Hessian approximation when $\|s_k\|$ is small compared to $\|y_k\|$, since it may be the case that $s_k^T y_k \gg s_k^T B_k s_k$.

Coleman and Conn [7] introduce an intermediate point, $\bar{x}_k \equiv x_k + Z_k p_z$, where p_z is the null space component of the step from x_k to x_{k+1}, defined in (2.4). They then define

$$s_k = Z_k^T(\bar{x}_k - x_k),$$
$$y_k = Z_k^T(\nabla_x L(\bar{x}_k, \lambda_k) - \nabla_x L(x_k, \lambda_k)).$$

This procedure ensures that $\|s_k\|$ cannot be too small compared to $\|y_k\|$, but requires an additional set of gradient evaluations at each iteration. This approach is the one used in most algorithms that are based on maintaining a positive-definite reduced Hessian matrix, such as those found in [8], [5], and [13].

Nocedal and Overton suggest an alternative strategy. If the null-space step is too small compared to the range-space step, they simply do not perform the update. Let

$$v_k = Y_k^T(x_{k+1} - x_k). \tag{2.9}$$

Update rule 1. Update B_k using the DFP or BFGS update formula if and only if the following rule holds:

$$\|v_k\| < \frac{\eta_1}{(k+1)^{1+\nu_1}}\|s_k\|, \tag{2.10}$$

where η_1 and ν_1 are appropriately chosen constants.

Consider now the Broyden update of D_k (2.7). The matrices B_k and D_k together provide an approximation to $Z^T W$. Therefore, both updates, when performed in tandem, should satisfy

$$Z_{k+1}^T W(x_{k+1} - x_k) = y_k.$$

So we can view the two updates,

$$D_{k+1}\hat{s}_k = y_k - B_k p_z \quad \text{and} \quad B_{k+1}s_k = y_k$$

as being done at the same time. We define $\hat{s}_k$ and $\hat{y}_k$ by

$$\hat{s}_k = v_k, \tag{2.11}$$
$$\hat{y}_k = y_k - B_k p_z, \tag{2.12}$$

where y_k and v_k are defined by (2.8) and (2.9).

In this case as well, the update may become unbounded as the solution is approached and is undefined when $\hat{s}_k = 0$. In fact, for many practical problems, $\|\hat{s}_k\|$ is very small so that we say that tangential convergence is observed and the update (2.7) would be undefined even far from the solution.

The $Z^T W Y$ update can be treated in the same manner that is used by Nocedal and Overton for the $Z^T W Z$ update. The relative sizes of the null space and range space components s_k and v_k are measured and the $Z^T W Y$ update is not performed if v_k is too small compared with s_k.

Update rule 2. Update D_k using the Broyden update if and only if the following rule holds:

$$\|s_k\| < \frac{\eta_2}{(k+1)^{1+\nu_2}}\|v_k\|. \tag{2.13}$$

The algorithm is now defined below. It is assumed that the initial point, x_0, is sufficiently close to x_* so that full steps are sufficient for convergence.

ALGORITHM 2.1 (local version).
Set x_0, B_0, D_0, tol.
Choose $\eta_1 > 0$, $\eta_2 > 0$, $\nu_1 > 0$, $\nu_2 > 0$.
Evaluate f_0, g_0, c_0, A_0.
Factor $A_0 = [Y_0 \quad Z_0]\begin{bmatrix} R_0 \\ 0 \end{bmatrix}$.
Set $k \leftarrow 0$.
While $\left\| \begin{bmatrix} Z_k^T g_k \\ c_k \end{bmatrix} \right\| \geq$ tol do
  Solve $R_k^T(p_Y) = -c_k$.
  Solve $B_k(p_Z) = -Z_k^T g_k - D_k p_Y$.
  Set $x_{k+1} = x_k + Y_k p_Y + Z_k p_Z$.
  Evaluate f_{k+1}, g_{k+1}, c_{k+1}, A_{k+1}.
  Factor $A_{k+1} = [Y_{k+1} \quad Z_{k+1}]\begin{bmatrix} R_{k+1} \\ 0 \end{bmatrix}$.
  Update B_k to form B_{k+1} using update rule 1.
  Update D_k to form D_{k+1} using update rule 2.
  Set $k \leftarrow k + 1$.
Endwhile

3. Local convergence. The convergence analysis presented in this section closely follows the results of Nocedal and Overton. These results are based on the Broyden, Dennis, and Moré theory [3] for unconstrained optimization. The results of this section show that given any ν_1 and ν_2 for (2.10) and (2.13), there exist $\varepsilon > 0$, $\delta_1 > 0$, $\delta_2 > 0$, $\eta_1 > 0$, and $\eta_2 > 0$, such that when η_1 and η_2 are used to define (2.10) and (2.13), x_0 is within ε in norm of x_* and the initial approximations B_0 and D_0 are within δ_1 and δ_2 in norm of the projected Hessian at x_*, then: (a) the sequence of iterates $\{x_k\}$ generated by Algorithm 2.1 converges to x_*; (b) if at least one of the updates to the matrices B_k or D_k is performed at each iteration, the convergence rate is Q-superlinear; and (c), if no assumptions are made regarding the updates being performed, the convergence rate is at least two-step Q-superlinear.

Assumptions. Assume that there is a local solution x_* to (1.1), with $A_* \equiv A(x_*)$ having full column rank, λ_* satisfying (2.2a), and the reduced Hessian $Z(x_*)^T W(x_*, \lambda_*) Z(x_*)$ positive-definite. It is assumed that $Y(x)$ and $Z(x)$ are Lipschitz continuous in the region near x_* and that $\nabla^2 f$ and $\{\nabla^2 c_i\}$ are Lipschitz continuous matrix functions. Assume further that $Z(x_*)^T W(x_*, \lambda_*) Y(x_*)$ is bounded and that $Z(x_*)^T W(x_*, \lambda_*)$ has full rank.

Notation. Let $e_k = x_k - x_*$, let $H_* = Z_*^T W_* Z_*$, and let $G_* = Z_*^T W_* Y_*$. The notation * denotes evaluation at (x_*, λ_*). Similarly, the subscript k is used to denote quantities evaluated at (x_k, λ_k). The notation $\|.\|$ is used for the Euclidean vector norm and the corresponding induced matrix norm. The Frobenius norm of an $m \times n$ matrix X given by $(\sum_{i=1}^m \sum_{j=1}^n x_{ij}^2)^{1/2}$ is denoted by $\|X\|_F$. The weighted Frobenius norm denoted by $\|X\|_M$, where X and M are $n \times n$ matrices and M is nonsingular, is defined by $(\text{trace}(MXM)^T(MXM))^{1/2}$.

The following inequalities are derived by Nocedal and Overton [27, p. 835] for $\|e_k\|$, $\|e_{k+1}\|$ small enough:

$$\|Z_k^T(g_{k+1} - A_{k+1}\lambda_{k+1}) - Z_k^T g_k - Z_k^T W_k(x_{k+1} - x_k)\| \leq C\|x_{k+1} - x_k\|^2, \tag{3.1}$$

$$\|Z_k^T g_k - Z_k^T W_k e_k\| \leq C\|e_k\|^2 \tag{3.2}$$

for a constant C.

By definition of the algorithm, the following must always hold:

$$Y_k^T(x_{k+1} - x_k) = -R_k^{-T} c_k, \tag{3.3}$$

$$Z_k^T(x_{k+1} - x_k) = -B_k^{-1} Z_k^T g_k + B_k^{-1} D_k R_k^{-T} c_k. \tag{3.4}$$

In the following, we develop a characterization of one-step Q-superlinear convergence for two-piece updates of the form used in Algorithm 2.1. Note, however, that there is a substantial change from the results of Nocedal and Overton. As a result of using an approximation to the "one-sided" projected Hessian, we are now able to derive bounds for $\|e_{k+1}\|$ in terms of $\|e_k\|$, instead of $\|e_{k-1}\|$. These bounds are used later in proving Q-linear convergence.

We first cite a lemma of Nocedal and Overton to show the contraction of certain quantities. We do not repeat the proof here, but note that the proof depends on the relation (3.3) and is not affected by the manner in which $Z_k^T W_k$ is approximated.

LEMMA 3.1 [27, Lemma 4.1, p. 835]. *Assume that for given k, $\|e_k\|$, and $\|e_{k-1}\|$ are sufficiently small. Then there exists a constant C_0, independent of the value of k, such that*

$$\text{(i)} \qquad \|Y_k^T e_{k+1}\| \le C_0 \|e_k\|^2,$$

$$\text{(ii)} \qquad \|Y_k^T e_k\| \le C_0(\|e_k\|^2 + \|e_{k-1}\|^2),$$

$$\text{(iii)} \qquad \|Y_k^T(x_{k+1} - x_k)\| \le C_0(\|e_k\|^2 + \|e_{k-1}\|^2).$$

THEOREM 3.1. *Suppose that Algorithm 2.1 is applied with any update rules. Let $\bar{\kappa}$ be a given constant. Then there exists an $\varepsilon_1 > 0$ such that, for any iteration k, if $\|B_{k-1}^{-1}\| \le \bar{\kappa}$, $\|B_k^{-1}\| \le \bar{\kappa}$, $\|D_k\| \le \bar{\kappa}$ and $\|e_{k-1}\| \le \varepsilon_1$, then*

$$\text{(i)} \qquad \|e_k\| \le C_1 \|e_{k-1}\|,$$

$$\text{(ii)} \qquad \|e_{k+1}\| \le C_1(\|e_k\|^2 + \|(B_k - H_*)Z_k^T e_k\| + \|(D_k - G_*)Y_k^T e_k\|),$$

$$\text{(iii)} \quad \|e_{k+1}\| \le C_1(\|e_k\|^2 + \|(B_k - H_*)Z_k^T(x_{k+1} - x_k)\| + \|(D_k - G_*)Y_k^T(x_{k+1} - x_k)\|),$$

where C_1 is some constant > 0.

Proof. We begin by noting that from (3.3) and (3.4)

$$\begin{aligned}\|e_k\| &\le \|x_k - x_{k-1}\| + \|e_{k-1}\| \\ &\le \|Y_{k-1} R_{k-1}^{-T} c_{k-1} + Z_{k-1} B_{k-1}^{-1} Z_{k-1}^T g_{k-1} + Z_{k-1} B_{k-1}^{-1} D_{k-1} R_{k-1}^{-T} c_{k-1}\| + \|e_{k-1}\|.\end{aligned}$$

The last term $\|e_{k-1}\|$ is ignored in the results of Nocedal and Overton, but is used here. The proof of (i) follows as in Nocedal and Overton from the boundedness of $\|R_{k-1}^{-T}\|$ and the fact that c and $Z^T g$ are Lipschitz continuous functions with $c_* = 0$ and $Z_*^T g_* = 0$.

To prove (ii), we use (3.2) and the orthogonality of $[Y_k \; Z_k]$, which gives

$$\|Z_k^T g_k - Z_k^T W_k(Z_k Z_k^T + Y_k Y_k^T) e_k\| \le C\|e_k\|^2.$$

Using the Lipschitz continuity of W, Z, and Y, this gives

$$\|Z_k^T g_k - H_* Z_k^T e_k - G_* Y_k^T e_k\| \le \tilde{C}_1 \|e_k\|^2 \tag{3.5}$$

for some constant $\tilde{C}_1$. Thus,

$$\begin{aligned}\|Z_k^T g_k - H_* Z_k^T e_k + [B_k Z_k^T e_k - B_k Z_k^T x_k + B_k Z_k^T x_*] \\ - [B_k Z_k^T e_{k+1} - B_k Z_k^T x_{k+1} + B_k Z_k^T x_*] \\ - G_* Y_k^T e_k + [D_k Y_k^T e_k - D_k Y_k^T x_k + D_k Y_k^T x_*] \\ - [D_k Y_k^T e_{k+1} - D_k Y_k^T x_{k+1} + D_k Y_k^T x_*]\| \le \tilde{C}_1 \|e_k\|^2,\end{aligned}$$

and therefore,

$$\begin{aligned}\|Z_k^T g_k - (H_* - B_k) Z_k^T e_k - B_k Z_k^T (e_{k+1} - (x_{k+1} - x_k)) - (G_* - D_k) Y_k^T e_k \\ - D_k Y_k^T (e_{k+1} - (x_{k+1} - x_k))\| \le \tilde{C}_1 \|e_k\|^2.\end{aligned}$$

From (3.4), we have that

$$B_k Z_k^T (x_{k+1} - x_k) = -Z_k^T g_k - D_k Y_k^T (x_{k+1} - x_k),$$

so (3.5) is equivalent to

$$\begin{aligned}&\| -(H_* - B_k) Z_k^T e_k - B_k Z_k^T e_{k+1} - (G_* - D_k) Y_k^T e_k - D_k Y_k^T e_{k+1}\| \le \tilde{C}_1 \|e_k\|^2, \\ &\|B_k Z_k^T e_{k+1} + D_k Y_k^T e_{k+1} - (B_k - H_*) Z_k^T e_k - (D_k - G_*) Y_k^T e_k\| \le \tilde{C}_1 \|e_k\|^2, \\ &\|B_k Z_k^T e_{k+1} + D_k Y_k^T e_{k+1}\| - \|(B_k - H_*) Z_k^T e_k + (D_k - G_*) Y_k^T e_k\| \le \tilde{C}_1 \|e_k\|^2, \\ &\|B_k Z_k^T e_{k+1} + D_k Y_k^T e_{k+1}\| \le \tilde{C}_1 \|e_k\|^2 + \|(B_k - H_*) Z_k^T e_k + (D_k - G_*) Y_k^T e_k\|.\end{aligned}$$

Using Lemma 3.1(i) and the fact that D_k is bounded, we have

$$\|B_k Z_k^T e_{k+1}\| \le \tilde{C}_2 \|e_k\|^2 + \|(B_k - H_*) Z_k^T e_k + (D_k - G_*) Y_k^T e_k\|.$$

Since B_k is positive-definite and B_k^{-1} is bounded,

$$\begin{bmatrix} B_k Z_k^T \\ Y_k^T \end{bmatrix}$$

is nonsingular and its inverse is bounded. Therefore,

$$\|e_{k+1}\| \le \left\| \begin{bmatrix} B_k Z_k^T \\ Y_k^T \end{bmatrix}^{-1} \right\| (\tilde{C}_3 \|e_k\|^2 + \|(B_k - H_*) Z_k^T e_k + (D_k - G_*) Y_k^T e_k\|),$$

which gives (ii).

Similarly, from (3.5), it follows that

$$\begin{aligned}\|Z_k^T g_k - H_* Z_k^T e_{k+1} + (H_* - B_k) Z_k^T (x_{k+1} - x_k) + B_k Z_k^T (x_{k+1} - x_k) \\ - G_* Y_k^T e_{k+1} + (G_* - D_k) Y_k^T (x_{k+1} - x_k) + D_k Y_k^T (x_{k+1} - x_k)\| \le \tilde{C}_1 \|e_k\|^2.\end{aligned}$$

Using (3.4) again, this gives

$$\begin{aligned}&\|H_* Z_k^T e_{k+1} + G_* Y_k^T e_{k+1}\| \\ &\quad \le \tilde{C}_1 \|e_k\|^2 + \|(B_k - H_*) Z_k^T (x_{k+1} - x_k) + (D_k - G_*) Y_k^T (x_{k+1} - x_k)\|.\end{aligned}$$

Lemma 3.1(i) and the boundedness of G_*, H_* and H_*^{-1} then gives (iii). □

Since Theorem 3.1 provides bounds for $\|e_{k+1}\|$ in terms of $\|e_k\|$, we are able to derive a characterization of one-step Q-superlinear convergence for Algorithm 2.1.

COROLLARY 3.1. *Suppose that Algorithm* 2.1 *is applied with any update rules for updating* B_k *and* D_k. *If* $x_k \to x_*$, $\|B_k\|$, $\|B_k^{-1}\|$, $\|D_k\| \le \bar{\kappa}$ *for all* k,

$$\omega_{1_k} \equiv \frac{\|(B_k - H_*)Z_k^T(x_{k+1} - x_k)\|}{\|(x_{k+1} - x_k)\|} \to 0$$

and

$$\omega_{2_k} \equiv \frac{\|(D_k - G_*)Y_k^T(x_{k+1} - x_k)\|}{\|(x_{k+1} - x_k)\|} \to 0.$$

then $x_k \to x_*$ *at a one-step* Q*-superlinear rate.*

Proof. From (iii) of Theorem 3.1, it follows that for k large enough,

$$\begin{aligned}\|e_{k+1}\| &\le C_1(\|e_k\|^2 + \omega_{1_k}\|x_{k+1} - x_k\| + \omega_{2_k}\|x_{k+1} - x_k\|) \\ &\le C_1(\|e_k\|^2 + \omega_{1_k}\|e_{k+1} - e_k\| + \omega_{2_k}\|e_{k+1} - e_k\|)\end{aligned}$$

and from Theorem 3.1(i),

$$\le C_1(\|e_k\|^2 + (1 + C_1)(\omega_{1_k}\|e_k\| + \omega_{2_k}\|e_k\|)).$$

Thus

$$\lim_{k\to\infty} \frac{\|e_{k+1}\|}{\|e_k\|} = \frac{C_1(\|e_k\|^2 + (1 + C_1)(\omega_{1_k}\|e_k\| + \omega_{2_k}\|e_k\|))}{\|e_k\|} \to 0. \qquad \square$$

Some of the results of Nocedal and Overton regarding the update of B_k are stated below without proof. These are stated for the DFP update (2.5), but can be modified for the BFGS update (2.6). Each of the theorems of Nocedal and Overton is paired with an analogous result that we prove for the update of D_k. By extending the theory developed by Nocedal and Overton we are able to prove local linear convergence for the new method (Theorem 3.3). In addition, we show that this two-piece updating method can achieve one-step Q-superlinear convergence when at least one of the matrices B_k or D_k is updated at each iteration (Theorem 3.4), and two-step Q-superlinear convergence if no assumptions are made regarding whether the updates are performed (Theorem 3.5).

LEMMA 3.2(a) [27, p. 837]. *Suppose that update rule* 1 *is used in Algorithm* 2.1. *If the update formula for* B_k *is used at iteration* k, *i.e.*, (2.10) *holds, then*

$$\|(x_{k+1} - x_k)\| \le (1 + \eta_1)\|s_k\|. \tag{3.6}$$

LEMMA 3.2(b). *Suppose that update rule* 2 *is used in Algorithm* 2.1. *If the update formula for* D_k *is used at iteration* k, *i.e.*, (2.13) *holds, then*

$$\|(x_{k+1} - x_k)\| \le (1 + \eta_2)\|v_k\|. \tag{3.7}$$

Proof. Let s_k and v_k be defined as in (2.8), (2.9). Then

$$\begin{aligned}\|(x_{k+1} - x_k)\| &\le \|Y_k Y_k^T(x_{k+1} - x_k)\| + \|Z_k Z_k^T(x_{k+1} - x_k)\| \\ &\le \|v_k\| + \|s_k\|\end{aligned}$$

and the result follows from (2.13). $\square$

THEOREM 3.2(a) [27, p. 837]. *Let* s_k, y_k *and* v_k *be defined by* (2.8), (2.9). *Then there exists an* $\varepsilon_{2a} > 0$ *such that if* $\|e_k\| \le \varepsilon_{2a}$, $\|e_{k+1}\| < \varepsilon_{2a}$, *then*

$$\|y_k - H_* s_k\| \le C_{2a} \max(\|e_k\|, \|e_{k+1}\|)\|(x_{k+1} - x_k)\| + \|G_* v_k\|. \tag{3.8}$$

THEOREM 3.2(b). *Let s_k, y_k and v_k be defined by* (2.8), (2.9) *and let $\hat{y}_k$ be defined by* (2.12). *Then there exists an $\varepsilon_{2b} > 0$ such that if $\|e_k\| \leq \varepsilon_{2b}$, $\|e_{k+1}\| < \varepsilon_{2b}$, then*

$$\|\hat{y}_k - G_* v_k\| \leq C_{2b} \max(\|e_k\|, \|e_{k+1}\|)\|(x_{k+1} - x_k)\| + \|(H_* - B_k)s_k\|, \tag{3.9}$$

where C_{2b} is a constant independent of k.

Proof. From the definition of $\hat{y}_k$ (2.12) and (2.8b) and v_k (2.9), we have

$$\|\hat{y}_k - G_* v_k\| = \|Z_k^T(g_{k+1} - A_{k+1}\lambda_{k+1}) - Z_k^T g_k - B_k Z_k^T(x_{k+1} - x_k) - G_* Y_k^T(x_{k+1} - x_k)\|.$$

Now, from (3.1), for ε_{2b} small enough,

$$\|Z_k^T(g_{k+1} - A_{k+1}\lambda_{k+1}) - Z_k^T g_k - Z_k^T W_k(Z_k Z_k^T + Y_k Y_k^T)(x_{k+1} - x_k)\| \leq C\|(x_{k+1} - x_k)\|^2$$

and thus

$$\begin{aligned}
&\|\hat{y}_k - G_* v_k\| \\
&\quad \leq \tilde{C}(\|(x_{k+1} - x_k)\|^2 + \|e_k\|\,\|(x_{k+1} - x_k)\| + \|(H_* - B_k)Z_k^T(x_{k+1} - x_k)\|) \\
&\quad \leq C_{2b} \max(\|e_k\|, \|e_{k+1}\|)\|(x_{k+1} - x_k)\| + \|(H_* - B_k)s_k\|
\end{aligned}$$

for some constants $\tilde{C}$ and C_{2b}. □

COROLLARY 3.2(a) [27, p. 837]. *Suppose that update rule* 1 *is used with the* DFP *update formula. Let $M \equiv H_*^{-1/2}$ and define*

$$\gamma_{1_k} \equiv \frac{\|y_k - H_* s_k\|}{\|s_k\|}. \tag{3.10}$$

Then, if $\max(\|e_k\|, \|e_{k+1}\|) \leq \varepsilon_{2a}$ *of Theorem* 3.2(a), *and if* (2.10) *holds at iteration k, then*

$$\frac{\|My_k - M^{-1}s_k\|}{\|M^{-1}s_k\|} \leq \|M\|^2 \gamma_{1_k} \tag{3.11}$$

and

$$\gamma_{1_k} \leq \left\{ C_{2a}(1 + \eta_1)\max(\|e_k\|, \|e_{k+1}\|) + \|G_*\|\frac{\|v_k\|}{\|s_k\|} \right\}. \tag{3.12}$$

COROLLARY 3.2(b). *Suppose that update rule* 2 *is used with the Broyden update formula. Define*

$$\gamma_{2_k} \equiv \frac{\|\hat{y}_k - G_* v_k\|}{\|v_k\|}. \tag{3.13}$$

Then if $\max(\|e_k\|, \|e_{k+1}\|) \leq \varepsilon_{2b}$ *of Theorem* 3.2(b), *and if* (2.13) *holds at iteration k, then*

$$\gamma_{2_k} \leq \left\{ C_{2b}(1 + \eta_2)\max(\|e_k\|, \|e_{k+1}\|) + \|(H_* - B_k)\|\frac{\|s_k\|}{\|v_k\|} \right\}. \tag{3.14}$$

Proof. The result follows from Lemma 3.2(b) and Theorem 3.2(b). □

LEMMA 3.3(a) [27, p. 838], based on [3, pp. 233, 242]. *Recall that $\|.\|_M$ denotes the weighted Frobenius norm defined above, and let M be any nonsingular symmetric matrix such that*

$$\frac{\|My_k - M^{-1}s_k\|}{\|M^{-1}s_k\|} \leq \frac{1}{3}, \quad \text{with } s_k \neq 0. \tag{3.15}$$

If the update formula (2.5) *is used at iteration* k, *then* $y_k^T s_k > 0$ *and*

$$\|B_{k+1} - H_*\|_M \leq \left((1 - \alpha\theta_{1_k}^2)^{1/2} + \frac{\alpha_1 \|My_k - M^{-1}s_k\|}{\|M^{-1}s_k\|}\right) \|B_k - H_*\|_M + \frac{\alpha_2 \|y_k - H_* s_k\|}{\|s_k\|}, \tag{3.16}$$

where $\alpha \in (0, 1]$, α_1, *and* α_2 *are constants independent of* k, *and*

$$1 \geq \theta_{1_k} = \begin{cases} \dfrac{\|M(B_k - H_*)s_k\|}{\|B_k - H_*\|_M \, \|M^{-1}s_k\|}, & B_k \neq H_*, \\ 0, & B_k = H_*. \end{cases}$$

LEMMA 3.3(b). *If the update formula* (2.7) *is used at iteration* k, *then, using the Frobenius norm,* $\|.\|_F$,

$$\|D_{k+1} - G_*\|_F \leq (1 - \alpha_3\theta_{2_k}^2)^{1/2}\|D_k - G_*\|_F + \frac{2\|\hat{y}_k - G_* v_k\|}{\|v_k\|}, \tag{3.17}$$

where $\alpha_3 \in (0, 1]$ *and*

$$1 \geq \theta_{2_k} = \begin{cases} \dfrac{\|(D_k - G_*)v_k\|}{\|D_k - G_*\|_F \, \|v_k\|}, & D_k \neq G_*, \\ 0, & D_k = G_*. \end{cases}$$

Proof. The proof follows from Broyden, Dennis, and Moré [3, pp. 233, 234]. □

In the following, we present a local linear convergence theorem that closely follows the theory developed by Nocedal and Overton. The crucial difference here arises from the assumption that D_0 is sufficiently close to $Z_*^T W_* Y_*$.

THEOREM 3.3. *Suppose that update rules* 1 *and* 2 *are used in Algorithm* 2.1. *Choose values* $\nu_1 > 0$ *and* $\nu_2 > 0$ *in conditions* (2.10) *and* (2.13). *Then for all* $r \in (0, 1)$, *there exists* $\varepsilon > 0$, $\delta_1 > 0$, $\delta_2 > 0$, $\eta_1 > 0$, *and* $\eta_2 > 0$, *such that if* $\|e_0\| \leq \varepsilon$, $\|B_0 - H_*\|_M \leq \delta_1$, $\|D_0 - G_*\|_F \leq \delta_2$, *and* η_1, η_2 *are used to define conditions* (2.10), (2.13), *then*

$$\|e_{k+1}\| \leq r\|e_k\|, \qquad k \geq 0, \tag{3.18}$$

i.e., $x_k \to x_*$ *at least at a* Q-*linear rate.*

Proof. Let α, α_1, and α_2 be the constants from Lemma 3.3(a) and let α_3 be the constant from Lemma 3.3(b). Define γ_{1_k} and M as in Corollary 3.2(a) and γ_{2_k} as in Corollary 3.2(b). Let $\bar{\alpha}_1 = \|M\|^2\alpha_1$. Let β_1 be such that $\|X\| \leq \beta_1\|X\|_M$, for any $t \times t$ matrix X, and β_2 be such that $\|\bar{X}\| \leq \beta_2\|\bar{X}\|_F$, for any $t \times m$ matrix $\bar{X}$. Let $\kappa = 2\|H_*^{-1}\|$ and $\bar{\delta} = 1/(2\|H_*^{-1}\|)$, so that, by the Banach Perturbation Lemma [28, p. 45],

$$\|X - H_*\| \leq \bar{\delta} \leftrightarrow \|X^{-1}\| \leq \kappa.$$

Let $\phi_1 = \|H_*\|$ and $\phi_2 = \|G_*\|$. Define $\varepsilon_2 \equiv \min(\varepsilon_{2a}, \varepsilon_{2b})$, where ε_{2a} and ε_{2b} are as in Theorems 3.2(a) and 3.2(b). Let ε_1, C_1 be as in Theorem 3.1, C_{2a} and C_{2b} be as in Theorems 3.2(a) and 3.2(b). Now require that $\varepsilon > 0$, $\delta_1 > 0$, $\delta_2 > 0$, $\eta_1 > 0$, and $\eta_2 > 0$ to be chosen small enough that

$$\varepsilon \leq \varepsilon_1/C_1, \tag{3.19a}$$

(3.19b) $\varepsilon \le \varepsilon_2/C_1$,

(3.19c) $\delta_1 \le \bar{\delta}/(2\beta_1)$,

(3.19d) $C_1\varepsilon(\varepsilon + \beta_1\delta_1 + \beta_2\delta_2) \le r < 1$,

(3.19e) $C_1(C_1\varepsilon + 2\beta_1\delta_1 + 2\beta_2\delta_2) \le r < 1$,

(3.19f) $\|M\|^2\{C_{2a}(1+\eta_1)C_1\varepsilon + \eta_1\|G_*\|\} \le \dfrac{1}{3}$,

(3.19g) $(2\bar{\alpha}_1\delta_1 + \alpha_2)\left(\dfrac{2C_{2a}(1+\eta_1)C_1\varepsilon}{1-r} + \eta_1\left(1+\dfrac{1}{\nu_1}\right)\|G_*\|\right) \le \delta_1$,

(3.19h) $2\left[\left(\dfrac{C_{2b}(1+\eta_2)C_1\varepsilon}{1-r}\right) + \eta_2\left(1+\dfrac{1}{\nu_2}\right)2\beta_1\delta_1\right] \le \delta_2$.

Such a choice of parameters is possible. Suppose $\|e_0\| \le \varepsilon$, $\|B_0 - H_*\|_M \le \delta_1$, and $\|D_0 - G_*\|_F \le \delta_2$. Define $\bar{\kappa} = \max(\kappa, \bar{\delta} + \phi_1, 2\beta_2\delta_2 + \phi_2)$. It will be established by induction that

(3.20i) $\|B_j - H_*\|_M \le 2\delta_1, \qquad j \ge 0.$

(3.20ii) $\|D_j - G_*\|_F \le 2\delta_2, \qquad j \ge 0.$

(3.20iii) $\|e_{j+1}\| \le r\|e_j\| \qquad j \ge 0.$

If update formula (2.5) is used at iteration j, then

(3.20iv) $$\begin{aligned}&\|B_{j+1} - H_*\|_M \\ &\quad \le [(1-\alpha\theta_{1_j}^2)^{1/2} + \bar{\alpha}_1\gamma_{1_j}]\|B_j - H_*\|_M + \alpha_2\gamma_{1_j}, \qquad \text{else } B_{j+1} = B_j.\end{aligned}$$

If update formula (2.7) is used at iteration j, then

(3.20v) $\|D_{j+1} - G_*\|_F \le (1-\alpha_3\theta_{2_j}^2)^{1/2}\|D_j - G_*\|_F + 2\gamma_{2_j}, \qquad \text{else } D_{j+1} = D_j.$

For $j = 0$, (3.20i) and (3.20ii) are true by assumption.

To prove (3.20iii) for $j = 0$, we note that since $\|B_0 - H_*\|_M \le \delta_1$, then $\|B_0 - H_*\| \le \beta_1\delta_1$, and therefore $\|B_0\| \le \bar{\delta} + \phi_1$ and $\|B_0^{-1}\| \le \kappa$ by (3.19c). We have also $\|D_0 - G_*\|_F \le \delta_2$, so that $\|D_0 - G_*\| \le \beta_2\delta_2$, which gives $\|D_0\| \le \beta_2\delta_2 + \phi_2$. Thus, with $\bar{\kappa}$ as defined above, Theorem 3.1(ii) can be applied, giving

$$\|e_1\| \le C_1(\|e_0\|^2 + \|(B_0 - H_*)Z_0^T e_0\| + \|(D_0 - G_*)Y_0^T e_0\|).$$

From the assumptions it follows that

$$\|e_1\| \le C_1\varepsilon(\varepsilon + \beta_1\delta_1 + \beta_2\delta_2)$$

so that the result follows from (3.19d).

The proof of (3.20iv) for $j = 0$ is given in [27, p. 839] and is repeated here for completeness. The inequality has only to be verified in the case where the update is used to obtain B_1. Since $\|B_0 - H_*\| \le \beta_1\delta_1$, it follows that $\|B_0\| \le \bar{\delta} + \phi_1$ and from (3.19c) $\|B_0^{-1}\| \le \kappa$. We have also $\|D_0 - G_*\|_F \le \delta_2$, which gives $\|D_0\| \le \beta_2\delta_2 + \phi_2$. Thus, with $\bar{\kappa}$ as defined above and $\|e_0\| \le \varepsilon \le \varepsilon_1$ from (3.19a), Theorem 3.1 applies, giving $\|e_1\| \le C_1\|e_0\| \le \varepsilon_2 \le \varepsilon_{2a}$ by (3.19b). Corollary 3.2(a) applies, since we also have $\|e_0\| \le \varepsilon_2$, which gives

$$\begin{aligned}\frac{\|My_0 - M^{-1}s_0\|}{\|M^{-1}s_0\|} &\le \|M\|^2\gamma_{1_0} \le (C_{2a}(1+\eta_1)C_1\varepsilon + \|G_*\|\eta_1)\|M\|^2 \\ &\le \frac{1}{3} \quad \text{by choice of } \varepsilon \text{ and } \eta_1 \text{ in (3.19f).}\end{aligned}$$

The proof of (3.20v) for $j = 0$ follows from Lemma 3.3(b).

Now, assuming (3.20i)–(3.20v) are true for $j = 0, 1, \ldots, k-1$, it is shown that they hold for $j = k$. The proofs of (3.20i) and (3.20ii) are quite similar, although the result (3.20ii) depends on using (3.20i). We refer to [27, p. 839] for the proof of (3.20i) and present the proof for (3.20ii). From (3.20ii) and (3.20v) it follows, regardless of whether the update formula (2.7) is used at iteration j, that

$$\|D_{j+1} - G_*\|_F - \|D_j - G_*\|_F \leq 2\gamma_{2_j}, \qquad j = 0, \ldots, k-1.$$

Since $\|e_0\| \leq \varepsilon$ and $\|e_1\| \leq C_1\varepsilon \leq \varepsilon_2 \leq \varepsilon_{2b}$, then, with (3.20iii), $j = 0, \ldots, k-1$, it follows that

$$\|e_i\| \leq r^i C_1 \varepsilon, \qquad i = 0, 1, \ldots, k. \tag{3.21}$$

Thus Corollary 3.2(b) applies for $j = 0, 1, \ldots, k-1$, which gives bounds on $\{\gamma_{2_j}\}$,

$$\gamma_{2_j} \leq \left\{ C_{2b}(1+\eta_2)C_1 r^i \varepsilon + \frac{\|(H_* - B_j)\|\,\|s_j\|}{\|v_j\|} \right\},$$

and thus

$$\|D_{j+1} - G_*\|_F - \|D_j - G_*\|_F \leq 2\left[C_{2b}(1+\eta_2)C_1 r^i \varepsilon + \frac{\|(H_* - B_j)\|\,\|s_j\|}{\|v_j\|} \right]$$

$$\|D_{j+1} - G_*\|_F - \|D_j - G_*\|_F \leq 2\left[C_{2b}(1+\eta_2)C_1 r^i \varepsilon + \frac{\eta_2}{(j+1)^{1+\nu_2}}\|(H_* - B_j)\| \right] \quad \text{for } j = 0, \ldots, k-1,$$

where condition (2.13) has been applied if the update formula is used at iteration j. When the update (2.7) is not used, $D_{j+1} = D_j$ and hence the inequality holds. From (3.20i), we know that $\|B_j - H_*\|_M \leq 2\delta_1$ for $j = 0, \ldots, k-1$ so that

$$\|B_j - H_*\| \leq 2\beta_1\delta_1. \tag{3.22}$$

Summing from $j = 0, 1, \ldots, k-1$,

$$\|D_k - G_*\|_F \leq \|D_0 - G_*\|_F + 2\left[\frac{C_{2b}(1+\eta_2)C_1\varepsilon}{1-r} + \eta_2\left(1 + \frac{1}{\nu_2}\right)(2\beta_1\delta_1) \right] \leq 2\delta_2$$

by choice of ε, δ_2, η_2 in (3.19). This proves (3.20ii) for $j = k$.

To prove (3.20iii), note that $\|e_{k-1}\| \leq C_1\varepsilon \leq \varepsilon_1$ from (3.21) and (3.19a), $\|B_{k-1} - H_*\| \leq 2\beta_1\delta_1$, $\|B_k - H_*\| \leq 2\beta_1\delta_1$ from (3.20i), and $\|D_{k-1} - G_*\| \leq 2\beta_2\delta_2$, $\|D_k - G_*\| \leq 2\beta_2\delta_2$ from (3.20ii). Therefore, $\|B_k^{-1}\|$, $\|B_{k-1}^{-1}\| < \kappa$, $\|B_k\| \leq \bar{\delta} + \phi_1$, and $\|D_k\| \leq 2\beta_2\delta_2 + \phi_2$. Thus, using $\bar{\kappa}$ as defined above, Theorem 3.1 can be applied, giving

$$\begin{aligned}
\|e_{k+1}\| &\leq C_1(\|e_k^2\| + 2\beta_1\delta_1\|e_k\| + 2\beta_2\delta_2\|e_k\|) \\
&\leq C_1(\|e_k\| + 2\beta_1\delta_1 + 2\beta_2\delta_2)\|e_k\| \\
&\leq C_1(C_1\varepsilon + 2\beta_1\delta_1 + 2\beta_2\delta_2)\|e_k\| \\
&\leq r\|e_k\|
\end{aligned}$$

by (3.19e) and (3.21). Note that here δ_2 is small by assumption, but that for Nocedal and Overton it may not be. This key assumption leads to Q-linear convergence.

The proof of (3.20iv) is almost the same as the proof of (3.20iv) for the case of $j = 0$. As above, we repeat the proof given in [27, p. 840]. As for the case $j = 0$, the inequality needs to be verified only in the event that the update is performed. We have $\|e_k\| \leq C_1\varepsilon \leq \varepsilon_2$ by (3.21) and (3.19b) and $\|e_{k+1}\| \leq \|e_k\| \leq C_1\varepsilon \leq \varepsilon_2$ by (3.20iii), (3.21), and (3.19b). Thus Corollary 3.2(a) applies, giving

$$\frac{\|My_k - M^{-1}s_k\|}{\|M^{-1}s_k\|} \leq \|M\|^2\gamma_{1_k} \leq (C_{2a}(1+\eta_1)C_1\varepsilon + \|G_*\|\eta_1)\|M\|^2 \leq \frac{1}{3} \text{ by (3.19f).}$$

Therefore, Lemma 3.3(a) applies, giving (3.20iv). Finally, (3.20v) follows from Lemma 3.3(b). □

At any given iteration, updates may be done for both B_k and D_k, only one update performed, or both updates skipped. In the following we show that as long as at least one of the updates is performed at each iteration, the local rate of convergence will be one-step Q-superlinear. However, if both updates are sometimes skipped, then the new method is actually very similar to the Nocedal–Overton method with skipped updates and is also two-step Q-superlinear. We note that it is indeed possible for the method to skip both updates, say if v and s are of the same size and $\eta_1, \eta_2 < 1$. The values of η_1, η_2 are constrained by (3.19) and could be small in theory. However, in practice, we have found that moderate values generally allow updating to be performed and yield superlinear convergence.

THEOREM 3.4. *Suppose that update rules* 1 *and* 2 *are used in Algorithm* 2.1. *Let* ε, δ_1, δ_2, η_1, *and* η_2 *be chosen as in Theorem* 3.3, *using any value* $r \in (0,1)$. *If at least one of the matrices* B_k *or* D_k *is updated at each iteration, then the rate of convergence is one-step* Q*-superlinear.*

Proof. Assume that $x_k \neq x^*$ for all k, since otherwise the result holds trivially. The goal is to show that

$$\zeta_k \equiv \frac{\|e_{k+1}\|}{\|e_k\|} \to 0.$$

Following the technique of Nocedal and Overton, we consider these two subsets of the iterates:

$$S_1 = \{k \mid \text{update rule (2.10) holds}\},$$
$$S_2 = \{k \mid \text{update rule (2.13) holds}\}.$$

Note that S_1 and S_2 need not be disjoint, since at early iterations both updates may be done. However, as $k \to \infty$, k will not be a member of both S_1 and S_2.

Recall from Theorem 3.1(iii) that

$$\|e_{k+1}\| \leq C_1(\|e_k\|^2 + \|(B_k - H_*)Z_k^T(x_{k+1} - x_k)\| + \|(D_k - G_*)Y_k^T(x_{k+1} - x_k)\|). \tag{3.23}$$

Define ω_{1_k} and ω_{2_k} as in Corollary 3.1, and suppose that s_k and v_k are defined by (2.8) and (2.9).

We first consider $\{\zeta_k \mid k \in S_1\}$, assuming that this subsequence is not finite. Let $S_1 = \{k_i \mid i = 1, 2, 3, \ldots\}$. We begin by reviewing the proof of Nocedal and Overton [27, p. 841] to show that $\omega_{1_{k_i}} \to 0$ as $i \to \infty$. Let

$$\rho_i = \|B_{k_i} - H_*\|_M,$$
$$\sigma_i = \tfrac{1}{2}\alpha\theta_{1_{k_i}}^2,$$
$$\bar{\gamma}_i = \gamma_{1_{k_i}}.$$

Note that $(1-\alpha\theta_{1_{k_i}}^2)^{1/2} \le 1-\sigma_i$. From (3.20iv), we have

$$\|B_{k_i+1}-H_*\|_M \le [(1-\alpha\theta_{1_{k_i}}^2)^{1/2}+\bar{\alpha}_1\gamma_{1_{k_i}}]\|B_{k_i}-H_*\|_M+\alpha_2\gamma_{1_{k_i}}. \tag{3.24}$$

Using Corollary 3.2(a), (3.21), and (2.10),

$$\gamma_{1_{k_i}} \le C_{2a}(1+\eta_1)C_1 r^{k_i}\varepsilon+\frac{\eta_1\|G_*\|}{(k_i+1)^{1+\nu_1}}.$$

Note that

$$\sum_i \gamma_{1_{k_i}} < \infty.$$

Because the update formula (2.5) or (2.6) is used at iteration k if and only if $k \in S_1$, we have

$$B_{k+1} = B_k \quad \text{for } k \notin S_1,$$

which means that

$$B_{k_{i+1}} = B_{(k_i+1)}.$$

Thus we have from (3.24) and the definitions of ρ_i, σ_i, and $\bar{\gamma}_i$,

$$\rho_{i+1} \le (1-\sigma_i+\bar{\alpha}_1\bar{\gamma}_i)\rho_i+\alpha_2\bar{\gamma}_i, \quad \text{with } \sum_i \bar{\gamma}_i < \infty. \tag{3.25}$$

Using a lemma of Han [21], Nocedal and Overton show that $\{\rho_i\}$ converges to a limit. By rearranging (3.25) to form

$$\rho_i\sigma_i \le \rho_i-\rho_{i+1}+(\bar{\alpha}_i\rho_i+\alpha_2)\bar{\gamma}_i,$$

and taking limits, we obtain $\rho_i\sigma_i \to 0$. Hence either $\rho_i \to 0$ or $\sigma_i \to 0$ as $i \to \infty$, so that in either case we obtain

$$\omega_{1_{k_i}} \to 0 \quad \text{as } i \to \infty. \tag{3.26}$$

Now, using (3.23) with (3.20ii), we have for i large enough,

$$\|e_{k_i+1}\| \le C_1(\|e_{k_i}\|^2+\omega_{1_{k_i}}\|x_{k_i+1}-x_{k_i}\|+2\beta_2\delta_2\|v_{k_i}\|).$$

Since update rule (2.10) holds for $k_i \in S_1$, we can bound $\|v_{k_i}\|$, resulting in

$$\begin{aligned}\|e_{k_i+1}\| &\le C_1\left(\|e_{k_i}\|^2+\omega_{1_{k_i}}\|x_{k_i+1}-x_{k_i}\|+2\beta_2\delta_2\frac{\eta_1}{(k_i+1)^{1+\nu_1}}\|Z_{k_i}^T(x_{k_i}^T(x_{k_i+1}-x_{k_i})\|\right)\\ &\le C_1\left(\|e_{k_i}\|^2+\omega_{1_{k_i}}\|x_{k_i+1}-x_{k_i}\|+2\beta_2\delta_2\frac{\eta_1}{(k_i+1)^{1+\nu_1}}\|x_{k_i+1}-x_{k_i}\|\right)\\ &\le C_1\left(\|e_{k_i}\|^2+\left[\omega_{1_{k_i}}+2\beta_2\delta_2\frac{\eta_1}{(k_i+1)^{1+\nu_1}}\right]\|x_{k_i+1}-x_{k_i}\|\right).\end{aligned}$$

Using (3.20iii), we obtain

$$\le C_1\left(\|e_{k_i}\|^2+\left[\omega_{1_{k_i}}+2\beta_2\delta_2\frac{\eta_1}{(k_i+1)^{1+\nu_1}}\right](1+r)\|e_{k_i}\|\right).$$

Thus, from (3.21), setting $\tilde{C}_1 = 2\beta_2\delta_2\eta_1$,

$$\frac{\|e_{k_i+1}\|}{\|e_{k_i}\|} \leq C_1\left(C_1 r^{k_i}\varepsilon + \left(\omega_{1_{k_i}} + \frac{\tilde{C}_1}{(k_i+1)^{1+\nu_1}}\right)(1+r)\right).$$

Thus, using (3.26), we have

$$\zeta_{k_i} = \frac{\|e_{k_i+1}\|}{\|e_{k_i}\|} \to 0 \quad \text{as } i \to \infty. \tag{3.27}$$

Now we turn to $\{\zeta_k \mid k \in S_2\}$, again assuming that this subsequence is not finite. The proof is very similar to the one above. Let $S_2 = \{k_j \mid j = 1, 2, 3, \ldots\}$. We first show that $\omega_{2_{k_j}} \to 0$ as $j \to \infty$, using the same techniques as before. Let

$$\begin{aligned} \bar{\rho}_j &= \|D_{k_j} - G_*\|_F, \\ \bar{\sigma}_j &= \tfrac{1}{2}\alpha_3\theta^2_{2_{k_j}}, \\ \tilde{\gamma}_j &= \gamma_{2_{k_j}}. \end{aligned}$$

Note that $0 \leq (1 - \alpha_3\theta^2_{2_{k_j}})^{1/2} \leq 1 - \bar{\sigma}_j \leq 1$. Using (3.20v), we have

$$\|D_{k_j+1} - G_*\|_F \leq (1 - \alpha_3\theta^2_{2_{k_j}})^{1/2}\|D_{k_j} - G_*\|_F + 2\gamma_{2_{k_j}}.$$

We now use Corollary 3.2(b) to bound $\gamma_{2_{k_j}}$, along with (3.21) and (2.13), giving

$$\gamma_{2_{k_j}} \leq C_{2b}(1+\eta_2)C_1 r^{k_j}\varepsilon + \frac{\eta_2\|H_* - B_{k_j}\|}{(k_j+1)^{1+\nu_2}}.$$

Therefore, using (3.22), we have

$$\sum_j \gamma_{2_{k_j}} < \infty.$$

Because the update formula (2.7) is used at iteration k if and only if $k \in S_2$, we have

$$D_{k_{j+1}} = D_{(k_j+1)}.$$

Thus we have

$$\bar{\rho}_{j+1} \leq (1 - \bar{\sigma}_j)\bar{\rho}_j + 2\tilde{\gamma}_j, \quad \text{with} \sum_j \tilde{\gamma}_j < \infty, \tag{3.28}$$

so that $\{\bar{\rho}_j\}$ converges to a limit. Rearranging (3.28), we have

$$\bar{\rho}_j\bar{\sigma}_j \leq \bar{\rho}_j - \bar{\rho}_{j+1} + 2\tilde{\gamma}_j.$$

Taking limits, we find that $\bar{\rho}_j\bar{\sigma}_j \to 0$. Hence either $\bar{\rho}_j \to 0$ or $\bar{\sigma}_j \to 0$ as $j \to \infty$, and in either case we obtain

$$\omega_{2_{k_j}} \to 0 \quad \text{as } j \to \infty. \tag{3.29}$$

Now, using Theorem 3.1(iii) with (3.20i) and (2.8a), we have

$$\|e_{k_j+1}\| \leq C_1(\|e_{k_j}\|^2 + 2\beta_1\delta_1\|s_{k_j}\| + \omega_{2_{k_j}}\|x_{k_j+1} - x_{k_j}\|).$$

Since update rule (2.13) holds for $k \in S_2$, we can bound $\|s_{k_j}\|$, giving

$$\|e_{k_j+1}\| \le C_1 \left(\|e_{k_j}\|^2 + 2\beta_1\delta_1 \frac{\eta_2}{(k_j+1)^{1+\nu_2}} \|v_{k_j}\| + \omega_{2_{k_j}} \|x_{k_j+1} - x_{k_j}\| \right)$$
$$\le C_1 \left(\|e_{k_j}\|^2 + 2\beta_1\delta_1 \frac{\eta_2}{(k_j+1)^{1+\nu_2}} \|x_{k_j+1} - x_{k_j}\| + \omega_{2_{k_j}} \|x_{k_j+1} - x_{k_j}\| \right).$$

Using (3.20iii) we have

$$\le C_1 \left(\|e_{k_j}\|^2 + \left[2\beta_1\delta_1 \frac{\eta_2}{(k_j+1)^{1+\nu_2}} + \omega_{2_{k_j}} \right] (1+r)\|e_{k_j}\| \right).$$

Thus, from (3.21), setting $\tilde{C}_2 = 2\beta_1\delta_1\eta_2$,

$$\frac{\|e_{k_j+1}\|}{\|e_{k_j}\|} \le C_1 \left(C_1 r^{k_j}\varepsilon + \left(\frac{\tilde{C}_2}{(k_j+1)^{1+\nu_2}} + \omega_{2_{k_j}} \right) (1+r) \right).$$

Therefore, using (3.29), we have

$$\zeta_{k_j} = \frac{\|e_{k_j+1}\|}{\|e_{k_j}\|} \to 0 \quad \text{as } j \to \infty. \tag{3.30}$$

Combining (3.27) and (3.30), we have

$$\zeta_k = \frac{\|e_{k+1}\|}{\|e_k\|} \to 0 \quad \text{as } k \to \infty. \qquad \square$$

In the final theorem we show that even if updates are skipped, the algorithm achieves a two-step Q-superlinear rate of convergence. Essentially, if the Broyden update (2.7) is not performed, the new method cannot achieve a faster rate of convergence than the Nocedal–Overton algorithm.

THEOREM 3.5. *Suppose that update rules* 1 *and* 2 *are used in Algorithm* 2.1. *Let* ε, δ_1, δ_2, η_1, *and* η_2 *be chosen as in Theorem* 3.3, *using any value* $r \in (0,1)$. *Then, in addition to the result* (3.18), *we also have that the rate of convergence is two-step* Q*-superlinear.*

Proof. We consider subsets of the iterates as in the proof of Theorem 3.4. In the proof of Theorem 3.4, we defined the sets S_1 and S_2. Let us now turn to the subset of the iterates defined by

$$S_3 = \{k \mid \text{neither update rule (2.10) nor update rule (2.13) hold}\}.$$

Let $S_3 = \{k_i \mid i = 1, 2, 3, \ldots\}$, and assume that this subsequence is not finite. Note that if S_3 is finite, then the results of Theorem 3.4 would apply. Using Theorem 3.1(iii),

$$\|e_{k_i+1}\| \le C_1(\|e_{k_i}\|^2 + \|B_{k_i} - H_*\|\|Z_{k_i}^T x_{k_i+1} - x_{k_i}\| + \|D_{k_i} - G_*\|\|Y_{k_i}^T x_{k_i+1} - x_{k_i}\|).$$

Using (3.20i), (3.20ii), and the fact that (2.10) does not hold at k_i, we obtain

$$\|e_{k_i+1}\| \le C_1 \left(\|e_{k_i}\|^2 + \left[\frac{2\beta_1\delta_1(k_i+1)^{1+\nu_1}}{\eta_1} + 2\beta_2\delta_2 \right] \|Y_{k_i}^T(x_{k_i+1} - x_{k_i})\| \right).$$

We can now apply Lemma 3.1(iii), giving,

$$\|e_{k_i+1}\| \le C_1 \left(\|e_{k_i}\|^2 + \left[\frac{2\beta_1\delta_1(k_i+1)^{1+\nu_1}}{\eta_1} + 2\beta_2\delta_2 \right] (C_0(\|e_{k_i}\|^2 + \|e_{k_i-1}\|^2)) \right).$$

Using Theorem 3.1(i), we obtain

$$\begin{aligned}&\|e_{k_i+1}\|\\&\quad\le C_1\left(C_1^2\|e_{k_i-1}\|^2+\left[\frac{2\beta_1\delta_1(k_i+1)^{1+\nu_1}}{\eta_1}+2\beta_2\delta_2\right](C_0(C_1^2\|e_{k_i-1}\|^2+\|e_{k_i-1}\|^2))\right).\\&\quad\le \tilde{C}(k_i+1)^{1+v_i}\|e_{k_i-1}\|^2,\quad \text{for some constant } \tilde{C}.\end{aligned}$$

Thus, by (3.21), setting $\bar{C}=C_1\tilde{C}\varepsilon$, we have

$$\zeta_{k_i}=\frac{\|e_{k_i+1}\|}{\|e_{k_i-1}\|}\le \bar{C}(k_i+1)^{1+\nu_1}r^{k_i}\to 0\quad \text{as } i\to\infty. \tag{3.31}$$

If we now combine (3.27), (3.30), and (3.31), we have

$$\zeta_k=\frac{\|e_{k+1}\|}{\|e_{k-1}\|}\to 0\quad \text{as } k\to\infty. \qquad \square$$

4. Numerical results. In this section the performance of Algorithm 2.1 is compared with the method of Nocedal and Overton on a small set of test problems. The programs were written in Fortran using double precision and run on an APOLLO 3500 workstation at Brooklyn College. The basic difference between these two methods is in the treatment of Z^TWY. It can be seen that on problems for which the Nocedal and Overton method clearly exhibits two-step convergence, the iterates produced by the new algorithm do not have the same behavior. In addition, the methods are comparable for problems on which the Nocedal–Overton method performs well.

Methods that update only the reduced Hessian often appear to be one-step superlinearly convergent in numerical experiments. It is generally difficult to find problems on which two-step but not one-step superlinear convergence is observed. Byrd [4] and Yuan [37] both describe problems on which methods that maintain only the reduced Hessian Z^TWZ cannot attain superlinear convergence, even when the exact reduced Hessian is available. The results presented in Tables 4.1–4.4 demonstrate that when an approximation to the Z^TWY matrix is available as in Algorithm 2.1, and used in addition to the exact reduced Hessian, then the method does achieve one-step superlinear convergence on the examples of Byrd and Yuan. The results presented in Tables 4.1–4.4 are not for the Nocedal–Overton algorithm and Algorithm 2.1 themselves, but for variants of these algorithms where the approximation to B_k is replaced by the exact reduced Hessian $Z_k^TW_kZ_k$. It should be noted that Byrd's analysis of two-step behavior was presented for the Coleman–Conn method rather than for the Nocedal–Overton

TABLE 4.1
Byrd $\mathbf{x}_0=(5.00,1.43)^{\mathbf{T}}$.

k	Nocedal–Overton using exact Z^TWZ	Algorithm 2.1 using exact Z^TWZ	Update
1	2.159e + 00	1.203e + 00	—
2	3.119e − 01	9.958e − 01	—
3	3.171e + 00	1.796e − 01	—
4	1.099e + 00	3.991e − 02	—
5	1.837e − 01	3.100e − 04	—
6	6.144e − 03	9.042e − 08	—
7	7.445e − 06	1.897e − 15	
8	1.108e − 11	—	

TABLE 4.2
Byrd $\mathbf{x}_0 = (5.00, 1.445)^{\mathbf{T}}$.

k	Nocedal–Overton using exact Z^TWZ	Algorithm 2.1 using exact Z^TWZ	Update
1	2.233e + 00	1.279e + 00	—
2	3.189e − 01	1.036e + 00	—
3	9.275e + 02	2.073e − 01	—
4	4.637e + 02	5.009e − 02	—
5	2.319e + 02	5.050e − 04	—
6	1.159e + 02	2.308e − 07	—
7	5.795e + 01	1.229e − 14	
8	2.895e + 01	—	
9	1.442e + 01	—	
10	7.104e + 00	—	
11	3.351e + 00	—	
12	1.343e + 00	—	
13	3.177e − 01	—	
14	2.002e − 02	—	
15	8.018e − 05	—	
16	1.286e − 09	—	

TABLE 4.3
Yuan $\mathbf{x}_0 = (.2, .1)^{\mathbf{T}}$.

k	Nocedal–Overton using exact Z^TWZ	Algorithm 2.1 using exact Z^TWZ	Update
1	2.281e − 01	1.882e − 01	D
2	1.662e − 01	4.406e − 02	D
3	8.181e − 02	7.646e − 03	—
4	6.551e − 02	5.501e − 04	—
5	4.921e − 03	1.615e − 05	—
6	4.914e − 03	6.231e − 08	—
7	2.440e − 05	4.590e − 11	—
8	2.439e − 05	8.469e − 15	—
9	5.952e − 10	4.575e − 21	—
10	5.948e − 10	1.558e − 28	
11	1.431e − 18	—	
12	3.537e − 19	—	

TABLE 4.4
Yuan $\mathbf{x}_0 = (.1, .1)^{\mathbf{T}}$.

k	Nocedal–Overton using exact Z^TWZ	Algorithm 2.1 using exact Z^TWZ	Update
1	9.000e − 02	1.204e − 01	D
2	9.000e − 02	2.024e − 02	—
3	9.900e − 03	8.421e − 04	—
4	9.900e − 03	1.757e − 05	—
5	9.999e − 05	2.610e − 08	—
6	9.999e − 05	1.386e − 11	—
7	1.000e − 08	1.316e − 15	—
8	1.000e − 08	3.753e − 22	—
9	1.000e − 16	3.382e − 30	
10	1.000e − 16	—	
11	3.607e − 32	—	
12	1.000e − 32	—	

method. However, for this problem $c(x_k) = c(x_k + Z_k s_k)$, so that the two algorithms coincide in this case.

Problem 1 (Byrd).

$$f(x) = \frac{1}{2}x_1^2 - 5x_1x_2 + \frac{1}{2}x_2^2 - \frac{(x_1-5)^3}{15}$$
$$\text{s.t. } c(x) = \frac{1}{(x_2-2)^2} - 1,$$
$$x_* = (5,1)^T.$$

Problem 2 (Yuan).

$$f(x) = \frac{x_2^2}{2} - x_1x_2 + \frac{-4(x_2-x_1)^3 - 6(x_2-x_1)^2(x_1-x_2^2) - 12(x_2-x_1)(x_1-x_2^2)^2 - 17(x_1-x_2^2)^3 + \dfrac{3(x_1-x_2^2)^4}{1-x_2}}{6(1-x_2)^3}$$
$$\text{s.t. } c(x) = x_1 + \frac{(x_2-x_1)^2 + (x_2-x_1)(x_1-x_2^2) + 2(x_1-x_2^2)^2}{(1-x_2)^2},$$
$$x_* = (0,0)^T.$$

In these cases, D_0 is defined by computing a finite-difference approximation to WZ, using a difference interval of 10^{-1} and multiplying $(WZ)^T Y$. The exact second derivatives are used for $Z^T WZ$ to compare with the results reported by Byrd and Yuan. The parameter values for the Broyden update were set to $\eta_2 = 1$ and $\nu_2 = 10^{-2}$; tol was set to 10^{-32}. A very small tolerance value was chosen to produce a larger sequence of iterates, so that the convergence pattern could be more clearly observed. The values reported in the tables are $\|x_k - x_{k-1}\|$.

The two-step behavior of reduced Hessian algorithms on the Byrd problem is sensitive to the starting point, x_0. In Table 4.1, this can be seen at the first five iterations, but eventually there is a tangential approach to the solution and convergence becomes one-step superlinear. Algorithm 2.1 can be seen to have one-step convergence from the start and achieves a better solution. The difference between the two methods is more clearly seen in Table 4.2. In this case, the new algorithm requires less than half as many iterations as the Nocedal–Overton method. In both cases, no updates of the D matrix were performed by Algorithm 2.1 for the chosen value of η_2. The faster convergence of Algorithm 2.1 is achieved because D_0 is close to $Z_*^T W_* Y_*$.

Tables 4.3 and 4.4 compare the performance of the Nocedal–Overton method with that of the new algorithm on Yuan's problem. The iterates produced by the Nocedal–Overton method exhibit a clear one-fast, one-slow pattern, while those produced by Algorithm 2.1 converge uniformly. In both cases the new method performed Broyden updates only at the first one or two iterations. For this example, if η_2 were to be set small enough that the Broyden updates are always skipped, the convergence of Algorithm 2.1 would be slightly slower.

As noted above, reduced Hessian methods often appear one-step superlinearly convergent. In practice, the steps generally become nearly tangent to the constraint normals as the solution is approached, which results in one-step superlinear convergence. Therefore, in general, it would not be expected that Algorithm 2.1 would converge significantly faster than the Nocedal–Overton method. In fact, it might be expected that the new method would result in a loss of efficiency, since it requires an extra $O((n-m)m)$ operations to perform the Broyden update. However, it appears that the use of update rule 2 results in skipping the Broyden update as soon as the tangential step becomes large. In the above examples, the Broyden update was

TABLE 4.5
Wright

k	N–O	Update	Algorithm 2.1	Updates
1	4.379e + 00	B	2.630e + 00	B-
2	7.514e + 00	B	4.520e + 00	B-
3	1.404e + 01	B	4.207e + 00	B-
4	1.519e + 01	B	3.487e + 00	B-
5	1.602e + 00	B	1.833e + 00	B-
6	1.086e + 00	B	7.280e − 01	-D
7	6.056e + 00	—	1.747e − 01	—
8	4.187e + 00	B	4.592e − 02	B-
9	2.260e + 00	—	1.024e − 01	B-
10	9.607e − 01	—	3.732e − 03	—
11	2.700e − 01	—	1.708e − 03	B-
12	2.545e − 02	—	1.528e − 05	B-
13	1.875e − 03	B	4.764e − 07	
14	4.976e − 03	B	—	
15	1.039e − 05		—	

hardly ever performed. Therefore, in practice, Algorithm 2.1 could exhibit behavior similar to the Nocedal–Overton method.

Tables 4.5–4.8 present the results of the two methods on a small set of test problems. In particular, these tables report on when the updates were performed. The entry B indicates an update of B_k, which approximates Z^TWZ; an entry of D indicates an update of D_k, the approximation to Z^TWY. In all cases, D_0 was defined using finite differences as described above and B_0 was defined in a similar manner.

The theory presented in §3 shows that given values of ν_1 and ν_2 for (2.10) and (2.13), there exist values of η_1, η_2, ε, δ_1, and δ_2 that will allow Algorithm 2.1 to attain one-step superlinear convergence. The values of these parameters are different for each problem since they are dependent on (3.19). For the results presented below, we chose one set of values for the parameters ν_1, ν_2, η_1, and η_2, and used those same values for all of the test problems. Since we did not adjust the parameters for each problem, it is not surprising that one-step superlinear convergence was not always seen for Algorithm 2.1. However, in practice one cannot necessarily determine the optimal values for the parameters and therefore our results are useful in comparing the behavior of the two algorithms.

In general, ν_1 and ν_2 must be small to allow summing the infinite series in the proof for (3.20i) and (3.20ii). The parameters η_1 and η_2 must be chosen large enough so that the updates are not skipped too often. However, if the η values are too large, the approximating matrices B_k and D_k may become unbounded. If the update to B_k is performed, there is not necessarily any benefit to also updating D_k. In fact, as $k \to \infty$, (2.10) and (2.13) cannot both be satisfied. However, if $\|v_k\|$ is large in comparison to $\|s_k\|$, it seems advantageous to update the D_k matrix even if B_k was also updated. Therefore, the η values were set with $\eta_1 > \eta_2$, so that the update to B_k is performed whenever possible and it is unlikely that D_k will be updated at the same time as B_k. The parameter values were set to $\eta_1 = 10$, $\nu_1 = 10^{-2}$, $\eta_2 = 1, \nu_2 = 10^{-2}$; tol was set to 10^{-8}. As before, the values reported are $\|x_k - x_{k-1}\|$.

Problem 3 (Wright [36]).

$$f(x) = x_1 x_2^2,$$
$$\text{s.t. } c(x) = x_1^2 + x_2^2 - 2,$$
$$x_* = (\sqrt{2}, 0)^T,$$
$$x_0 = (1.65, -.8165)^T.$$

TABLE 4.6
Hexagon

k	N–O	Update	Algorithm 2.1	Updates
1	1.834e − 01	—	1.834e − 01	-*D*
2	2.461e − 02	*B*	4.940e − 02	*B*-
3	8.437e − 03	*B*	3.811e − 02	*B*-
4	4.929e − 03	*B*	5.708e − 03	*B*-
5	2.182e − 03	*B*	5.161e − 03	*B*-
6	1.188e − 03	*B*	6.994e − 03	*B*-
7	6.652e − 04	*B*	1.919e − 03	*B*-
8	4.777e − 04	*B*	1.257e − 03	*B*-
9	6.739e − 04	*B*	3.378e − 04	*B*-
10	3.889e − 04	*B*	6.581e − 05	*B*-
11	6.691e − 05	*B*	1.277e − 05	*B*-
12	5.455e − 05	*B*	6.076e − 06	*B*-
13	7.055e − 05	*B*	7.680e − 07	*B*-
14	3.810e − 05	*B*	7.051e − 07	*B*-
15	1.254e − 05	*B*	1.363e − 07	
16	9.788e − 06	*B*	—	
17	1.246e − 05	*B*	—	
18	2.066e − 06	*B*	—	
19	1.955e − 06	*B*	—	
20	9.531e − 07	*B*	—	
21	9.335e − 08	*B*	—	
22	7.072e − 08	*B*	—	
23	1.240e − 07	*B*	—	
24	2.861e − 07	*B*	—	
25	1.877e − 07	*B*	—	
26	4.153e − 08	*B*	—	
27	1.775e − 07	*B*	—	
28	2.622e − 08		—	

TABLE 4.7
Hock/Schittkowski 39

k	N–O	Update	Algorithm 2.1	Updates
1	1.734e − 01	—	1.735e − 01	-*D*
2	3.756e − 02	—	3.751e − 02	-*D*
3	1.774e − 03	*B*	1.824e − 03	*B*-
4	2.706e − 04	*B*	3.578e − 04	*B*-
5	1.741e − 06	*B*	2.508e − 06	*B*-
6	7.017e − 09		3.353e − 07	*B*-
7	—		1.205e − 07	

Problem 4 (Hexagon). This problem is derived from an inequality-constrained problem described in [15]. For purposes of testing these methods, the problem was transformed into an equality constrained problem by eliminating the constraints that are not active at the solution.

$$x_0 = (.06095,\ .5976,\ 1.2,\ .5976,\ .06095,\ .3438,\ .501,\ -.501,\ -3438)^T.$$

Problems 5 *and* 6. These are problems 39 and 93 respectively in [23]. For Problem 5, $x_0 = (.91,\ .90,\ .003,\ .0015)^T$; for Problem 6, $x_0 = (5.28,\ 4.652,\ 10.43,\ 12.09,\ .745,\ .856)^T$.

With the exception of Hock/Schittkowski 39, the new algorithm required fewer iterations than the method of Nocedal and Overton. For the chosen value of η_2, updates to the D_k matrix were hardly ever performed. For Hock/Schittkowski 93, the update to D_0 was never

TABLE 4.8
Hock/Schittkowski 93

k	N–O	Update	Algorithm 2.1	Updates
1	6.958e − 02	—	7.078e − 02	B-
2	1.178e − 01	B	1.336e − 02	B-
3	2.169e − 02	B	4.868e − 03	B-
4	1.691e − 01	B	1.929e − 03	B-
5	6.803e − 02	B	9.749e − 05	B-
6	6.827e − 03	B	1.831e − 05	B-
7	8.614e − 03	B	3.019e − 06	B-
8	3.801e − 03	B	5.628e − 08	B-
9	9.753e − 04	B	8.702e − 09	
10	2.675e − 04	B	—	
11	1.853e − 05	B	—	
12	2.864e − 06	B	—	
13	4.241e − 08	B	—	
14	5.308e − 09		—	

performed. As noted above, the results of Algorithm 2.1 do not always show one-step superlinear convergence, although the convergence is generally faster than the Nocedal–Overton method. This may be due to the choice of values for the η and ν parameters, the fact that a line search is not performed, or to the initial coarse finite difference approximations to the projected Hessian.

In all of the test problems, the update to D_k was hardly ever performed. It appears that the faster convergence of Algorithm 2.1 is a result of D_0 being close to $Z_*^T W_* Y_*$. Since the local convergence is almost always tangential, $\|v_k\|$ is not generally greater than $\|s_k\|$, and the update to D_k is not needed. Further study is needed to investigate the performance of the new algorithm when started far from the solution. The numerical results clearly indicate that having a good approximation to $Z_*^T W_* Y_*$ results in faster local convergence. It is to be hoped that when x_0 is far from the solution, performing updates to D_k will build up a good approximation to $Z_*^T W_* Y_*$.

Although numerical tests have been reported for only a small number of problems, the results are encouraging. It appears that the new algorithm requires fewer iterations than the Nocedal–Overton method and, generally, does not require a substantial increase in computation. The convergence rate of Algorithm 2.1 is dependent upon the proper choice of values for the parameters η_1, η_2, ν_1 and ν_2, just as the convergence rate of the Nocedal–Overton method is dependent upon the proper choice of values for η_1 and ν_1. A topic for future research is to develop a method of automatically controlling the value of η_1 and η_2 so that the updates are not skipped unnecessarily and yet the approximating matrices do not become unbounded.

Acknowledgments. The author would like to thank Michael Overton for suggesting this work and both Jorge Nocedal and Michael Overton for carefully reading the manuscript and providing many helpful suggestions. The referees provided many constructive comments and corrections. Thanks also go to Peter Fenyes for several helpful conversations and to Noson Yanofsky for generating some of the test results.

REFERENCES

[1] M. C. BIGGS, *Constrained minimization using recursive equality quadratic programming*, in Numerical Methods for Non-Linear Optimization, F.A. Lootsma, ed., Academic Press, London, New York, 1972, pp. 411–428.

[2] P. T. BOGGS, J. W. TOLLE, AND P. WANG, *On the local convergence of quasi-Newton methods for constrained optimization*, SIAM J. Control Optim., 20 (1982), pp. 161–171.

[3] C. G. Broyden, J. E. Dennis, Jr., and J. J. Moré, *On the local and superlinear convergence of quasi-Newton methods*, J. Inst. Math. Appl., 12 (1973), pp. 223–245.
[4] R. H. Byrd, *An example of irregular convergence in some constrained optimization methods that use the projected Hessian*, Math. Programming, 32 (1985), pp. 232–237.
[5] R. H. Byrd and J. Nocedal, *An analysis of reduced Hessian methods for constrained optimization*, Math. Programming, 49 (1990), pp. 285–323.
[6] R. H. Byrd and R. B. Schnabel, *Continuity of the null space basis and constrained optimization*, Math. Programming, 35 (1986), pp. 32–41.
[7] T. F. Coleman and A. R. Conn, *On the local convergence of a quasi-Newton method for the nonlinear programming problem*, SIAM J. Numer. Anal., 21 (1984), pp. 755–769.
[8] T. F. Coleman and P. A. Fenyes, *Partitioned quasi-Newton methods for nonlinear equality constrained optimization*, Math. Programming, 53 (1992), pp. 17–44.
[9] T. F. Coleman and D. C. Sorensen, *A note on the computation of an orthonormal basis for the null space of a matrix*, Math. Programming, 29 (1984), pp. 234–242.
[10] J. E. Dennis and J. J. Moré, *Quasi-Newton methods, motivation and theory*, SIAM Rev., 19 (1977), pp. 46–89.
[11] R. Fletcher, *Practical Methods of Optimization, Vol. 2, Constrained Optimization*, John Wiley and Sons, New York, Toronto, 1981.
[12] R. Fontecilla, T. Steihaug, and R. A. Tapia, *A convergence theory for a class of quasi-Newton methods for constrained optimization*, SIAM J. Numer. Anal., 24 (1987), pp. 1133–1151.
[13] R. Fontecilla, *Local convergence of secant methods for nonlinear constrained optimization*, SIAM J. Numer. Anal., 25 (1988), pp. 692–712.
[14] D. Gabay, *Reduced quasi-Newton methods with feasibility improvement for nonlinear constrained optimization*, Math. Programming Study, 16 (1982), pp. 18–44.
[15] P. E. Gill, W. Murray, M. A. Saunders, and M. H. Wright, *Model building and practical aspects of nonlinear programming*, in Computational Mathematical Programming, K. Schittkowski, ed., NATO ASI Series F: Computer and Systems Sciences 15, Springer-Verlag, Berlin, New York, 1985, pp. 209–247.
[16] ———, *User's Guide for* NPSOL (*Version* 4.0): *A Fortran Package for Nonlinear Programming*, Tech. Report SOL 86-2, Systems Optimization Laboratory, Stanford University, Stanford, CA, 1986.
[17] P. E. Gill, W. Murray, G. W. Stewart, and M. H. Wright, *Properties of a basis for the null space*, Math. Programming, 33 (1985), pp. 172–186.
[18] S. T. Glad, *Properties of updating methods for the multipliers in augmented Lagrangians*, J. Optim.Theory Appl., 28 (1979), pp. 135–156.
[19] C. B. Gurwitz, *Sequential quadratic programming methods based on approximating a projected Hessian matrix*, Ph.D. thesis, Computer Science Department, New York University, New York, 1986.
[20] C. B. Gurwitz and M. L. Overton, *Sequential quadratic programming methods based on approximating a projected Hessian matrix*, SIAM J. Sci. Statist. Comput., 10 (1989), pp. 631–653.
[21] S.-P. Han, *Superlinearly convergent variable metric algorithms for general nonlinear programming problems*, Math. Programming, 11 (1976), pp. 263–282.
[22] ———, *A globally convergent method for nonlinear programming*, J. Optim. Theory Appl., 22 (1977), pp. 297–310.
[23] W. Hock and K. Schittkowski, *Test examples for nonlinear programming codes*, Lecture Notes in Economics and Mathematical Systems 187, Springer-Verlag, Berlin, New York, 1981.
[24] W. Murray, *An algorithm for constrained minimization*, in Optimization, R. Fletcher, ed., Academic Press, London, New York, 1969, pp. 247–258.
[25] W. Murray and M. H. Wright, *Projected Lagrangian methods based on the trajectories of penalty and barrier functions*, Systems Optimization Laboratory Report 78–23, Stanford University, Stanford, CA, 1978.
[26] ———, *Computation of the search direction in constrained optimization algorithms*, Math. Programming Study, 16 (1982), pp. 62–83.
[27] J. Nocedal and M. L. Overton, *Projected Hessian updating algorithms for nonlinearly constrained optimization*, SIAM J. Numer. Anal., 22 (1985), pp. 821–850.
[28] J. M. Ortega and W. C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, New York, London, 1970.
[29] 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, New York, London, 1978, pp. 27–63.
[30] K. Schittkowski, *The nonlinear programming method of Wilson, Han and Powell with an augmented Lagrangian type line search function*, Numer. Math., 38 (1981), pp. 83–114.
[31] F. Tagliaferro, *On a Quasi-Newton Update and its Application to Equality Constrained Optimization*, Universita di Trieste, Trieste, Italy, manuscript.
[32] K. Tanabe, *Feasibility-improving gradient-acute-projection methods: A unified approach to nonlinear programming*, Lecture Notes in Numer. Appl. Anal., 3 (1981), pp. 57–76.

[33] R. A. TAPIA, *Diagonalized multiplier methods and quasi-Newton methods for constrained optimization*, J. Optim. Theory Appl., 22 (1977), pp. 135–194.

[34] R. B. WILSON, *A Simplical Algorithm for Concave Programming*, Ph.D. thesis, Graduate School of Business Administration, Harvard University, Cambridge, MA, 1963.

[35] R. S. WOMERSLEY AND R. FLETCHER, *An algorithm for composite nonsmooth optimization problems*, J. Optim. Theory Appl. 48 (1986), pp. 493–523.

[36] M. H. WRIGHT, *Numerical Methods for Nonlinearly Constrained Optimization*, Ph.D. thesis, Stanford University, Stanford, CA, 1976.

[37] Y. YUAN, *On the reduced Hessian method for constrained optimization*, Math. Programming, 32 (1985), pp. 224–231.

SIAM J. OPTIMIZATION
Vol. 4, No. 3, pp. 486–511, August 1994
© 1994 Society for Industrial and Applied Mathematics
003

ON SMOOTHING EXACT PENALTY FUNCTIONS FOR CONVEX CONSTRAINED OPTIMIZATION*

MUSTAFA Ç. PINAR† AND STAVROS A. ZENIOS‡

Abstract. A quadratic smoothing approximation to nondifferentiable exact penalty functions for convex constrained optimization is proposed and its properties are established. The smoothing approximation is used as the basis of an algorithm for solving problems with (i) embedded network structures, and (ii) nonlinear minimax problems. Extensive numerical results with large-scale problems illustrate the efficiency of this approach.

Key words. convex constrained optimization, exact penalty functions, ϵ-smoothing, network problems with side constraints, minimax optimization

AMS subject classifications. 90C25, 90C06, 90B10

1. Introduction.

We consider the problem

$$\text{[NLP]} \qquad \underset{x \in \Re^n}{\text{minimize}} \quad f_0(x)$$

$$\text{subject to} \quad f_i(x) \leq 0 \quad \forall \quad i = 1, \ldots, K$$
$$g_i(x) \leq 0 \quad \forall \quad i = 1, \ldots, L.$$

The functions $f_i : \Re^n \longmapsto \Re$ and $g_i : \Re^n \longmapsto \Re$ are convex and at least once continuously differentiable. Let $X = \{x | x \in \Re^n, g_i(x) \leq 0 \text{ for all } i = 1, \ldots, L\}$, and assume that X is a compact subset of $\Re^n$. Furthermore, we make the following assumptions.

A1. [NLP] has a nonempty and compact optimal solution set;

A2. [NLP] has at least one feasible solution that satisfies all the constraints with strict inequality.

Using assumptions A1 and A2 it follows from [25, Coro. 28.2.1] that a Kuhn–Tucker vector (or a Lagrange multiplier vector as defined in [25, p. 274]) exists for [NLP].

Consider the convex penalty function $p : \Re \mapsto \Re$ such that

$$p(t) \begin{cases} > 0 & \text{for } t > 0 \\ = 0 & \text{otherwise,} \end{cases} \tag{1}$$

where t is a scalar variable. We want to obtain a solution to [NLP] by solving the following exact penalty problem:

$$\text{[EP]} \qquad \min_{x \in X} f_0(x) + \sum_{i=1}^{K} p(f_i(x)). \tag{2}$$

* Received by the editors July 22, 1991; accepted for publication (in revised form) April 6, 1993. This research was partially supported by National Science Foundation grant CCR-91-04042 and Air Force Office of Scientific Research grant 91-0168.

† Institute for Numerical Analysis, Technical University of Denmark, 2800 Lyngby, Denmark (nimp@unidhp.uni-c.dk).

‡ Decision Sciences Department, University of Pennsylvania, Philadelphia, Pennsylvania 19104 (zenios@wharton.upenn.edu).

In this paper we focus on the ℓ_1 penalty function

$$p(t) = \mu \max\{0, t\}, \tag{3}$$

where μ is a positive scalar that determines the slope of the function and the severity of the penalty. It is well known that under the assumptions stated above, the optimal solutions to [EP] and [NLP] coincide for a finite value of the penalty parameter μ; see, for instance, [2, Prop. 1]. We denote the exact penalty function for the constraints $f_i(x) \leq 0$ by

$$f_i^+(x) = \max\{0, f_i(x)\}.$$

The function p defined by (3) is not continuously differentiable at $t = 0$. To gain access to gradient based minimization techniques to solve [EP], we consider an ϵ-smoothing of the function p. Let the ϵ-smoothed function p_ϵ be the following:

$$p_\epsilon(t) = \begin{cases} 0 & \text{if } t \leq 0, \\ \mu \frac{t^2}{2\epsilon} & \text{if } 0 \leq t \leq \epsilon, \\ \mu(t - \frac{\epsilon}{2}) & \text{if } t \geq \epsilon, \end{cases} \tag{4}$$

where ϵ is a positive scalar; see Fig. 1. The following property can be easily verified:

$$\lim_{\epsilon \to 0} p_\epsilon(t) = p(t). \tag{5}$$

We note that the smooth penalty function introduced here can be obtained as a special case of the smoothing technique introduced by Bertsekas in [3]. Zang used a similar smoothing approximation in [31] for minimax problems extending the work in [3]. The smoothing technique introduced by Zang is symmetric around the kink while our smoothing approximation is asymmetric. The function p_ϵ is a linear-quadratic function that has been analyzed by Rockafellar in [26] and [27]. Interestingly, linear-quadratic functions are also used in statistics to develop robust estimation procedures. This function is known as the Huber estimator in computational statistics [11]. The linear-quadratic smoothing function was also used in [17] to develop a finite algorithm to solve ℓ_1 minimization problems that arise in linear regression.

Our motivation for considering the penalty approximation [EP] to the original [NLP] is the desire to exploit embedded structures in large-scale problems. Of particular interest is the class of network problems with side constraints. The classical multicommodity network flow is separated into simple network flow problems by placing the coupling constraints in the penalty function [23], [29]. The same is true for certain formulations of stochastic programming problems (i.e., the split variable formulation of [28]). Such problems can be decomposed into scenario subproblems by placing the "complicating" constraints in the penalty function. Our approach is also applicable to the robust optimization models of [19]. The introduction of the ϵ-smoothing device is made for practical reasons. An algorithm based on ϵ-smoothing is shown here to be very efficient and robust. Furthermore, the ϵ-smoothing device provides a natural mechanism for handling "soft" constraints, i.e., constraints that do not need to be satisfied exactly.

The ϵ-smoothed penalty function has been used in our earlier report [33] for the design of a numerical algorithm for the multicommodity network flow problem. The algorithm is also suitable for parallel implementation. A parallel implementation on a shared memory multiprocessor supercomputer CRAY Y-MP was developed in [23].

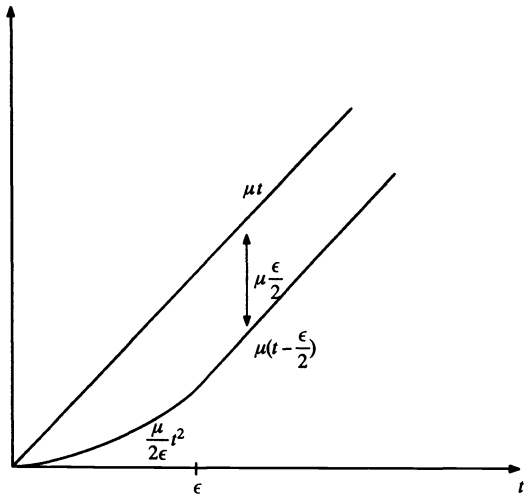


FIG. 1. *Linear-quadratic smooth penalty function.*

The goal of this paper is twofold. The first goal is to analyze the linear-quadratic penalty function as a *smooth* approximation to the ℓ_1 exact penalty function and derive bounds to the approximation error. Hence, we provide the analysis to support the experimental work reported in [23]. The second goal is to extend the application of the linear-quadratic penalty (LQP) algorithm to other optimization problems beyond the multicommodity network flows in [33]. We develop an implementation of the LQP algorithm to solve network flow problems with side constraints and use it to solve personnel management problems from the Navy Personnel Research Department and nonlinear embedded network problems drawn from econometric studies at the World Bank. The Navy personnel scheduling problem, in particular, defied solution by other network-based codes [14]; prior to the development of the algorithm reported here this model was solved using a heuristic method [15]. Another contribution of this paper is the development of an LQP algorithm for minimax optimization problems, and the empirical testing of this algorithm using large-scale minimax problems from the literature. Results with this problem class are also encouraging.

1.1. Preliminaries and organization. It is well known that a solution to a convex program can be obtained by solving the exact penalty problem. An exact solution is obtained if the penalty parameter is larger than a threshold value, characterized by the largest component of a Lagrange multiplier vector of the original problem. This assertion is equivalent to a nondifferentiability requirement on the penalty function as shown in [2]. This rules out the application of gradient based descent methods for the solution of the penalty problem. However, it is possible to obtain a Lagrange multiplier vector to the original problem in a finite number of minimizations even with a differentiable penalty function provided that the penalty function is "steep" enough and some special structure is present in the original problem. These conditions are made precise in [2] and illustrated with the quadratic penalty function. It is shown that for polyhedral convex programs it is possible to obtain a Lagrange multiplier vector to the original problem—provided that one exists—by solving a quadratic penalty problem. But with the quadratic penalty function, the characterization of the threshold value in terms of the Lagrange multiplier vector is lost. However, it is possible to design algorithms based on differentiable exact penalty functions [4], [9].

This paper contains the following results. First we examine the properties of the ϵ-smooth linear-quadratic penalty function as an approximation to the ℓ_1 exact penalty function. We show that an a priori upper bound to the approximation error can be computed as a function of the penalty parameters. It is shown that an optimum point to the original problem is not necessarily an optimum of the penalty problem even for values of the penalty parameter larger than the threshold value. However, its suboptimality can be bounded by a computable constant. This result is a partial generalization of a classical result that states that if the original problem is solvable, then the penalty problem is solvable and that the optima coincide provided that the penalty parameter is larger than a threshold. This result has been proved in various forms; see [2] for a list of references.

The main result of the paper states that it is possible to compute a solution that is feasible to within ϵ if the penalty parameter is specified as a constant multiple of the threshold value for the nondifferentiable case. Such a solution is termed *ϵ-feasible.* From the main result it follows that there is a unique threshold value of the penalty parameter for a given problem to achieve ϵ–feasibility for any ϵ. Furthermore, an upper bound on the distance from the optimal value of the original problem can be estimated.

For polyhedral convex programs we show that, due to the differentiability property, the solution to the smooth penalty problem yields the Lagrange multiplier vector assuming it is unique. Therefore it is possible to compute the Lagrange multiplier vector to a linear program in a finite number of minimizations. Thus we recover the precise characterization of the threshold value, which was well known for the nondifferentiable case, with a differentiable penalty function. These results are developed in §2.

In §3 we develop a simple algorithm motivated by the main result of the paper and give numerical examples. Sections 4 and 5 report computational results with (i) large-scale network-structured programs and (ii) nonlinear minimax problems.

2. A smooth approximation. In this section, we describe the properties of the linear-quadratic penalty function. We start with some elementary properties first. These properties are given in §2.1. In §2.2 we study the conditions under which the penalty problem yields an ϵ-feasible solution to the original problem.

2.1. Basic properties. Let us define the objective function for the exact penalty:

$$F(x,\mu) = f_0(x) + \sum_{i=1}^{K} p(f_i(x)); \tag{6}$$

and the objective function for the ϵ-smoothed penalty:

$$\tilde{F}(x,\mu,\epsilon) = f_0(x) + \sum_{i=1}^{K} p_\epsilon(f_i(x)). \tag{7}$$

We can now state the following result which gives an a priori upper bound on the smoothing approximation error.

LEMMA 1. *Let the functions F and $\tilde{F}$ be as defined by* (6) *and* (7). *Then*

$$0 \le F(x,\mu) - \tilde{F}(x,\mu,\epsilon) \le K\mu\frac{\epsilon}{2} \tag{8}$$

for any $x \in \Re^n$, $\mu > 0$ and $\epsilon > 0$.

Proof. Using the definitions of p and $\tilde{p}$, we get

$$0 \leq p(f_i(x)) - p_\epsilon(f_i(x)) \leq \mu\frac{\epsilon}{2} \quad \forall\ i = 1, \ldots, K \quad \text{and} \quad x \in \Re^n. \tag{9}$$

Adding up for all i, we obtain

$$0 \leq \sum_{i=1}^{K} p(f_i(x)) - \sum_{i=1}^{K} p_\epsilon(f_i(x)) \leq K\mu\frac{\epsilon}{2} \quad \forall\ x \in \Re^n. \tag{10}$$

The result immediately follows from the definitions of F and $\tilde{F}$. □

We propose to solve the following smoothed penalty problem

$$\text{[SEP]} \qquad \min_{x \in X} \tilde{F}(x, \mu, \epsilon) \tag{11}$$

instead of solving [EP]. Before we proceed we need the following definition.

DEFINITION 1. *A vector $\tilde{x} \in X$ is ϵ-feasible if*

$$f_i(\tilde{x}) \leq \epsilon \quad \forall\ i = 1, \ldots, K.$$

The following results give a priori upper bounds on the error incurred by solving the smooth penalty problem [SEP] in lieu of the nondifferentiable penalty problem [EP].

PROPOSITION 1. *Let $x^* \in X$ be an optimal solution of* [EP] *and $\tilde{x} \in X$ be an optimal solution of* [SEP] *for some μ and ϵ. Then*

$$0 \leq F(x^*, \mu) - \tilde{F}(\tilde{x}, \mu, \epsilon) \leq K\mu\frac{\epsilon}{2}. \tag{12}$$

Proof. From (8) we have

$$F(x, \mu) \leq \tilde{F}(x, \mu, \epsilon) + K\mu\frac{\epsilon}{2}. \tag{13}$$

Taking the infimum, we obtain

$$\inf_{x \in X} F(x, \mu) \leq \inf_{x \in X} \tilde{F}(x, \mu, \epsilon) + K\mu\frac{\epsilon}{2}, \tag{14}$$

which proves the right-hand inequality. The left-hand inequality can be proved similarly. □

PROPOSITION 2. *Let $x^* \in X$ be an optimal solution of* [EP] *and $\tilde{x} \in X$ be an optimal solution of* [SEP] *for some μ and ϵ. Furthermore, let x^* be feasible to* [NLP] *and $\tilde{x}$ be ϵ-feasible. Then*

$$0 \leq f_0(x^*) - f_0(\tilde{x}) \leq K\mu\epsilon. \tag{15}$$

Proof. By hypothesis, $\tilde{x}$ is ϵ-feasible, which implies the following:

$$\sum_{i=1}^{K} p_\epsilon(f_i(\tilde{x})) \leq K\mu\frac{\epsilon}{2}. \tag{16}$$

Again by hypothesis, x^* is a solution to **[NLP]**, which implies that

$$\sum_{i=1}^{K} p(f_i(x^*)) = 0. \tag{17}$$

From Proposition 1, we have

$$0 \leq f_0(x^*) + \sum_{i=1}^{K} p(f_i(x^*)) - f_0(\tilde{x}) - \sum_{i=1}^{K} p_\epsilon(f_i(x)) \leq K\mu\frac{\epsilon}{2}. \tag{18}$$

By substituting (16) and (17) into (18) and rearranging terms the result is established. □

Proposition 2 states that the difference between the optimal values of the original problem and the penalty problem can be controlled through the smoothing parameter ϵ if the solution to the penalty problem is ϵ-feasible. However, it does not specify conditions for the penalty parameter μ to achieve ϵ-feasibility. It is known (see, for instance, [2]) that for μ larger than a threshold μ^*, where $\mu^* > \max_i y_i^*$ and y^* is a Lagrange multiplier vector of the original problem [NLP], an optimal solution x^* to [NLP] is also an optimal solution for [EP]. This result provided the motivation for designing exact penalty methods [4]. The next question is: How is this result affected by the ϵ-smoothing of the exact penalty function? The answer, as we show, is that although an optimal solution to [NLP] does not necessarily coincide with an optimal solution to [SEP], its suboptimality can be bounded as a function of the penalty and smoothing parameters.

PROPOSITION 3. *Let x^* be an optimal solution for* [NLP] *and let $(y^*, z^*) \in \Re^{K+L}$ be a Lagrange multiplier vector. Then for some $\epsilon > 0$*

$$\tilde{F}(x^*, \mu, \epsilon) \leq \tilde{F}(x, \mu, \epsilon) + K\mu\frac{\epsilon}{2} \quad \forall \quad x \in X \tag{19}$$

provided that $\mu > y_i^*$ for $i = 1, \ldots, K$.

Proof. By convexity of f_i for $i = 0, \ldots, K$, we have

$$f_i(x) \geq f_i(x^*) + \nabla f_i(x^*)^T(x - x^*) \quad \forall \quad x \in X. \tag{20}$$

Since x^* is an optimal solution for [NLP] and (y^*, z^*) is a Lagrange multiplier vector, we have (see, e.g., [25, Thm. 28.3])

$$\nabla f_0(x^*) = -\sum_{i=1}^{K} y_i^* \nabla f_i(x^*) - \sum_{i=1}^{L} z_i^* \nabla g_i(x^*), \tag{21}$$

$$y_i^* f_i(x^*) = 0 \quad \forall \quad i = 1, \ldots, K, \qquad z_i^* g_i(x^*) = 0 \quad \forall \quad i = 1, \ldots, L, \tag{22}$$

$$y_i^* \geq 0 \quad \forall \quad i = 1, \ldots, K, \qquad z_i^* \geq 0 \quad \forall \quad i = 1, \ldots, L, \tag{23}$$

$$f_i(x^*) \leq 0 \quad \forall \quad i = 1, \ldots, K, \qquad g_i(x^*) \leq 0 \quad \forall \quad i = 1, \ldots, L. \tag{24}$$

Using (20)–(22) and the definition of F, we obtain

$$\begin{aligned} F(x, \mu) &\geq f_0(x^*) + \nabla f_0(x^*)^T(x - x^*) + \mu \sum_{i=1}^{K} f_i^+(x) \\ &= f_0(x^*) - \sum_{i=1}^{K} y_i^* \nabla f_i(x^*)^T(x - x^*) \end{aligned}$$

$$
\begin{aligned}
&\quad - \sum_{i=1}^{L} z_i^* \nabla g_i(x^*)^T (x - x^*) + \mu \sum_{i=1}^{K} f_i^+(x) \qquad \text{(using (21))} \\
&\geq f_0(x^*) - \sum_{i=1}^{K} y_i^* (f_i(x) - f_i(x^*)) \\
&\quad - \sum_{i=1}^{L} z_i^* (g_i(x) - g_i(x^*)) + \mu \sum_{i=1}^{K} f_i^+(x) \qquad \text{(using (20))} \\
&= f_0(x^*) - \sum_{i=1}^{K} y_i^* f_i(x) - \sum_{i=1}^{L} z_i^* g_i(x) + \mu \sum_{i=1}^{K} f_i^+(x) \qquad \text{(using (22))}.
\end{aligned}
$$

Since $f_i(x) \leq f_i^+(x)$, we have

$$
F(x, \mu) \geq f_0(x^*) + \sum_{i=1}^{K} (\mu - y_i^*) f_i^+(x) - \sum_{i=1}^{L} z_i^* g_i(x).
$$

Thus, for $\mu \geq y_i^*$ for all $i = 1, \ldots, K$, and observing that $z_i^* \geq 0$, and $g_i(x) \leq 0$ for all $i = 1, \ldots, L$ by hypothesis, we get

$$
F(x, \mu) \geq f_0(x^*). \tag{25}
$$

However, from Lemma 1, we have

$$
F(x, \mu) - \tilde{F}(x, \mu, \epsilon) \leq K\mu \frac{\epsilon}{2}. \tag{26}
$$

Rewriting (25) as

$$
f_0(x^*) - F(x, \mu) \leq 0 \tag{27}
$$

and observing that since x^* is also feasible, $f_0(x^*) = \tilde{F}(x^*, \mu, \epsilon)$. We then add up inequalities (26) and (27) to obtain

$$
\tilde{F}(x^*, \mu, \epsilon) - \tilde{F}(x, \mu, \epsilon) \leq K\mu \frac{\epsilon}{2}, \tag{28}
$$

which establishes the result. □

Therefore the suboptimality of x^* in [SEP] is bounded by the quantity $K\mu\frac{\epsilon}{2}$. It can be observed that as ϵ vanishes, one recovers the classical assertion that an optimal solution to the original problem coincides with an optimal solution to the penalty problem for a finite value of the penalty parameter μ. Thus, Proposition 3 generalizes Proposition 1 of Bertsekas [2] in this respect. An immediate corollary of Proposition 3 can be stated as follows.

COROLLARY 1. *Let $\bar{x}$ be a minimum point of*

$$
\min_{x \in X} \tilde{F}(x, \mu, \epsilon)
$$

and x^* be an optimal point for the original problem [NLP] *and let $y^* \in \Re^K$ be the Lagrange multiplier vector associated with the inequalities $f_i(x) \leq 0$ for all $i = 1, \ldots, K$. Then*

$$
0 \leq f_0(x^*) - \tilde{F}(\bar{x}, \mu, \epsilon) \leq K\mu \frac{\epsilon}{2} \tag{29}
$$

provided that $\mu > y_i^*$ *for* $i = 1, \ldots, K$.

Proposition 3, together with the following result that is identical to Proposition 3.5 of [31], also engenders an important consequence.

PROPOSITION 4. *Let* $\{\epsilon_k\} \to 0$ *be a sequence of positive numbers and assume that* x_k *is a solution to*

$$\min_{x \in X} \tilde{F}(x, \mu, \epsilon_k) \tag{30}$$

for some $\mu > 0$. *Also let* $\bar{x}$ *be an accumulation point of the sequence* $\{x_k\}$, *then*

$$F(\bar{x}, \mu) = F(x(\mu), \mu), \tag{31}$$

where $x(\mu)$ *is an optimal solution to*

$$\min_{x \in X} F(x, \mu).$$

Proof. The result follows directly from the continuity of F, (5) and (12). □

Therefore if the penalty parameter μ was specified larger than the threshold value μ^*, the solutions to [EP] and [NLP] would coincide. Hence, we could hope to obtain a solution to the original problem [NLP] in the limit by solving smooth penalty problems for decreasing smoothing parameters ϵ_k and increasing penalty parameters μ_k since the threshold value μ^* is not known in practice. Also, driving ϵ to 0 is not desirable since we recover the nondifferentiable ℓ_1 penalty function. On the other hand, we lose the "exactness" property by solving the smooth penalty problem instead of the nondifferentiable penalty problem. That is, we cannot expect to obtain a solution to the original problem by solving a linear-quadratic penalty problem. However, we can obtain an ϵ-feasible solution by solving a linear-quadratic penalty problem. Such a solution also provides a guaranteed error bound from the unknown optimal value of the original problem by virtue of Proposition 2. In §2.2 we show that ϵ-feasibility is attained if the penalty problem [SEP] is solved using a penalty parameter value equal to a constant multiple of the threshold value for the nondifferentiable ℓ_1 penalty function. This result motivates a simple algorithm.

2.2. Approximate or ϵ-exactness. In this section we study the approximate *exactness* properties that the smooth penalty function inherits from its nondifferentiable counterpart. We are particularly interested in achieving ϵ-feasibility. We begin with some observations regarding the consequences of ϵ-feasibility. We first study the optimality conditions of problems [NLP] and [SEP], respectively.

Optimality conditions for [NLP]. From [25, Thm. 28.3] we know that for x^* to be an optimal solution for [NLP], and for (y^*, z^*) to be a Lagrange multiplier vector, it is necessary and sufficient that (x^*, y^*, z^*) is a saddlepoint of the Lagrangian of [NLP] (as defined in [25, p. 280]). That is, if and only if the components of (x^*, y^*, z^*) satisfy the following conditions:

$$\nabla f_0(x^*) = -\sum_{i=1}^{K} y_i^* \nabla f_i(x^*) - \sum_{i=1}^{L} z_i^* \nabla g_i(x^*), \tag{32}$$

$$y_i^* f_i(x^*) = 0 \quad \forall \quad i = 1, \ldots, K, \qquad z_i^* g_i(x^*) = 0 \quad \forall \quad i = 1, \ldots, L, \tag{33}$$

$$y_i^* \geq 0 \quad \forall \quad i = 1, \ldots, K, \qquad z_i^* \geq 0 \quad \forall \quad i = 1, \ldots, L, \tag{34}$$

$$f_i(x^*) \leq 0 \quad \forall \quad i = 1, \ldots, K, \qquad g_i(x^*) \leq 0 \quad \forall \quad i = 1, \ldots, L. \tag{35}$$

The existence of (y^*, z^*) is guaranteed under assumption A2.

Optimality conditions for [SEP]. From [25, Thm. 28.3] we know that for x to be an optimal solution for [SEP], and for z to be a Lagrange multiplier vector, it is necessary and sufficient that (x, z) is a saddlepoint of the Lagrangian of [SEP] (as defined in [25, p. 280]). This condition holds if and only if the components of (x, z) satisfy the following conditions:

$$\nabla f_0(x) = -\sum_{i \in \mathcal{A}} \frac{\mu f_i^+(x)}{\epsilon} \nabla f_i(x) - \sum_{i \in \mathcal{V}} \mu \nabla f_i(x) - \sum_{i=1}^{L} z_i \nabla g_i(x), \tag{36}$$

$$z_i g_i(x) = 0 \quad \forall \quad i = 1, \ldots, L, \tag{37}$$

$$z_i \geq 0 \quad \forall \quad i = 1, \ldots, L. \tag{38}$$

The existence of z is guaranteed under assumption A2. Recall here that $f_i^+(x) = \max\{0, f_i(x)\}$. We use

$$\mathcal{A} = \{i | f_i(x) \leq \epsilon\}$$

to denote the set of indices corresponding to constraints that are satisfied or violated to within ϵ at x, and

$$\mathcal{V} = \{i | f_i(x) > \epsilon\}$$

for the set of indices corresponding to constraints violated beyond ϵ at x.

Now consider $(\tilde{x}, \tilde{z})$ satisfying conditions (36)–(38) for some μ and ϵ. The Lagrange multiplier estimate $\tilde{y}$ is obtained from

$$\tilde{y}_i = \frac{dp_\epsilon}{dt}(f_i(\tilde{x})) \quad \forall \quad i = 1, \ldots, K. \tag{39}$$

The optimality conditions to [SEP] closely resemble those of [NLP] under the observation that the triple $(\tilde{x}, \tilde{y}, \tilde{z})$ satisfies the optimality conditions to [NLP] with the exception of the complementary slackness and feasibility conditions. However, by assumption $\tilde{x}$ is ϵ-feasible. Hence,

$$f_i(\tilde{x}) \leq \epsilon \ \forall \ i = 1, \ldots, K.$$

It can be easily shown that the error in complementary slackness is bounded by the quantity $\mu\epsilon$ if $\tilde{x}$ is ϵ-feasible. To see this, observe that all constraints are satisfied to within ϵ and that the Lagrange multiplier estimates are given by

$$\tilde{y}_i = \frac{\mu \max(0, \tilde{u}_i)}{\epsilon} \quad \forall \quad i = 1, \ldots, K,$$

where

$$\tilde{u}_i = f_i(\tilde{x}).$$

Therefore we have

$$\tilde{y}_i \tilde{u}_i \leq \max\left\{0, \mu \frac{\tilde{u}_i^2}{\epsilon}\right\} \quad \forall \quad i = 1, \ldots, K.$$

However, since $\tilde{u}_i \leq \epsilon$ for $i = 1, \ldots, K$, the following holds:

$$\tilde{y}_i \tilde{u}_i \leq \mu\epsilon \quad \forall \quad i = 1, \ldots, K. \tag{40}$$

Hence, the assertion is verified.

The observations above motivate the effort to compute an ϵ-feasible solution to the original problem by solving the smooth penalty problem [SEP]. In the remainder of this section we study the conditions under which we can obtain a solution that is ϵ-feasible for the original problem by solving the linear-quadratic penalty problem. Truemper [30] showed that it is possible to attain ϵ-feasibility if the penalty parameter μ is chosen as a multiple of the threshold μ^*, where $\mu^* > \max_i y_i^*$ with the classical quadratic penalty function. We give a similar result using the linear-quadratic penalty function in Proposition 5. This result characterizes the threshold value of the penalty parameter μ and provides a sufficient condition for ϵ-feasibility.

PROPOSITION 5. *Let x^* be an optimal solution to* [NLP], *and (y^*, z^*) be a Lagrange multiplier vector. Let $\tilde{x}$ be an optimal solution to the smooth penalty problem* [SEP] *for some μ and ϵ. Then $\tilde{x}$ is ϵ-feasible if μ is chosen such that*

$$\mu = \frac{\mu^*}{\kappa}, \tag{41}$$

where κ is a constant given by

$$\kappa = \frac{1 - \sqrt{K}}{1 - K}. \tag{42}$$

Proof. Recall that $\mathcal{A}$ is the set of indices corresponding to constraints that are satisfied or violated to within ϵ at $\tilde{x}$, and $\mathcal{V}$ is the set of indices corresponding to constraints violated beyond ϵ at $\tilde{x}$. We assume that the cardinality of the set $\mathcal{V}$ is at least one and argue by contradiction. We can write the objective function of the smooth penalty problem [SEP] at the assumed minimum $\tilde{x}$ as

$$f_0(\tilde{x}) + \mu \sum_{i \in \mathcal{V}} \left(f_i(\tilde{x}) - \frac{\epsilon}{2}\right) + \mu \sum_{i \in \mathcal{A}} \frac{f_i^+(\tilde{x})^2}{2\epsilon}.$$

We consider the linear term first. The linear term can be rewritten as

$$\mu \sum_{i \in \mathcal{V}} \left(f_i^+(\tilde{x}) - \frac{\epsilon}{2}\right) = \mu^* \sum_{i \in \mathcal{V}} f_i^+(\tilde{x}) + \left(\mu \sum_{i \in \mathcal{V}} \left(f_i^+(\tilde{x}) - \frac{\epsilon}{2}\right) - \mu^* \sum_{i \in \mathcal{V}} f_i^+(\tilde{x})\right).$$

Considering the term in parentheses and recalling the definition of μ as a function of μ^*, we get

$$\mu \sum_{i \in \mathcal{V}} \left(f_i^+(\tilde{x}) - \frac{\epsilon}{2}\right) - \mu^* \sum_{i \in \mathcal{V}} f_i^+(\tilde{x}) = \mu^* \left(\sum_{i \in \mathcal{V}} \frac{(f_i^+(\tilde{x}) - \frac{\epsilon}{2})}{\kappa} - f_i^+(\tilde{x})\right).$$

Now we define $t = f_i^+(\tilde{x})$, the term in the summation becomes

$$\frac{t - \frac{\epsilon}{2}}{\kappa} - t.$$

Since $f_i(\tilde{x}) > \epsilon$ for all $i \in \mathcal{V}$,

$$\frac{t - \frac{\epsilon}{2}}{\kappa} - t > \frac{\epsilon}{2\kappa} - \epsilon.$$

By definition $\kappa < \frac{1}{2}$, which makes the right-hand side of the above inequality positive. Since the cardinality of the set $\mathcal{V}$ is at least one, we have

$$\mu \sum_{i \in \mathcal{V}} \left(f_i^+(\tilde{x}) - \frac{\epsilon}{2} \right) > \mu^* \sum_{i \in \mathcal{V}} f_i^+(\tilde{x}) + \mu^* \left(\frac{\epsilon}{2\kappa} - \epsilon \right). \tag{43}$$

Now we consider the quadratic term. The quadratic term can also be rewritten as

$$\mu \sum_{i \in \mathcal{A}} \frac{f_i^+(\tilde{x})^2}{2\epsilon} = \mu^* \sum_{i \in \mathcal{A}} f_i^+(\tilde{x}) + \left(\mu \sum_{i \in \mathcal{A}} \frac{f_i^+(\tilde{x})^2}{2\epsilon} - \mu^* \sum_{i \in \mathcal{A}} f_i^+(\tilde{x}) \right).$$

Again considering the term in parentheses,

$$\mu \sum_{i \in \mathcal{A}} \frac{f_i^+(\tilde{x})^2}{2\epsilon} - \mu^* \sum_{i \in \mathcal{A}} f_i^+(\tilde{x}) = \mu^* \sum_{i \in \mathcal{A}} \left(\frac{f_i^+(\tilde{x})^2}{2\epsilon\kappa} - f_i^+(\tilde{x}) \right).$$

Defining again $t = f_i^+(\tilde{x})$, we get the term in the summation as

$$\frac{t^2}{2\epsilon\kappa} - t,$$

which is a convex function in t. The minimum is attained at $t^* = \epsilon\kappa$—recall that $\kappa < \frac{1}{2}$—at which point the function takes the value $-\epsilon\kappa/2$. Since this lower bound is negative following the positivity of ϵ and κ, and since the cardinality of the set $\mathcal{A}$ is assumed less than $K - 1$, the following inequality can be written:

$$\mu \sum_{i \in \mathcal{A}} \frac{f_i(\tilde{x})^2}{2\epsilon} \geq \mu^* \sum_{i \in \mathcal{A}} f_i^+(\tilde{x}) - (K-1)\mu^* \left(\frac{\epsilon\kappa}{2} \right). \tag{44}$$

Now combining inequalities (43) and (44), we get

$$\begin{aligned}
& f_0(\tilde{x}) + \mu \sum_{i \in \mathcal{V}} \left(f_i(\tilde{x}) - \frac{\epsilon}{2} \right) + \mu \sum_{i \in \mathcal{A}} \frac{f_i^+(\tilde{x})^2}{2\epsilon} \\
& > f_0(\tilde{x}) + \mu^* \sum_{i=1}^{K} f_i^+(\tilde{x}) + \mu^* \left(\frac{\epsilon}{2\kappa} - \epsilon \right) - (K-1)\mu^* \left(\frac{\epsilon\kappa}{2} \right) \\
& = f_0(\tilde{x}) + \mu^* \sum_{i=1}^{K} f_i^+(\tilde{x}) \\
& > f_0(\tilde{x}) + \sum_{i=1}^{K} y_i^* f_i(\tilde{x}) + \sum_{i=1}^{L} z_i^* g_i(\tilde{x}) \\
& \geq f_0(x^*) + \sum_{i=1}^{K} y_i^* f_i(x^*) + \sum_{i=1}^{L} z_i^* g_i(x^*) \\
& = f_0(x^*) \\
& = f_0(x^*) + \sum_{i=1}^{K} p_\epsilon(f_i(x^*)).
\end{aligned}$$

The first equality follows since

$$\left(\frac{\epsilon}{2\kappa} - \epsilon\right) - (K-1)\left(\frac{\epsilon\kappa}{2}\right) = 0$$

by definition of κ. The second inequality follows from the definition of μ^*. The third inequality along with the second equality follow from the definition of a Lagrange multiplier vector. The last equality follows from the feasibility of x^*. But this gives a contradiction since $\tilde{x}$ was assumed to be an optimal solution to [SEP]. Hence, the set $\mathcal{V}$ must be empty. □

We remark that the constant κ in Truemper's result involves the parameter ϵ, whereas our constant does not. This is due to our smoothing device that already incorporates ϵ. Therefore, for a given problem, the threshold value of μ to attain ϵ-feasibility is a constant independent of ϵ. The existence of a threshold value for the penalty parameter μ indicates that an ϵ-feasible solution can be computed in a single or at most a finite number of minimizations. An important consequence of Proposition 5 is that we can now characterize the conditions under which the upper bound on the difference between the optimal values as stated in Proposition 2 of §2.1 is achieved. This result is now summarized in Proposition 6.

PROPOSITION 6. *Let x^* be an optimal solution for* [NLP] *and $y^* \in \Re^K$ be the Lagrange multiplier vector. Let μ^* be such that $\mu^* > y_i^*$ for all $i = 1, \ldots, K$. Furthermore, let $\tilde{x}$ be a solution to* [SEP]. *Then for $\mu = \mu^*/\kappa$.*

$$0 \leq f_0(x^*) - f_0(\tilde{x}) \leq K\mu\epsilon, \tag{45}$$

where

$$\kappa = \frac{1 - \sqrt{K}}{1 - K}.$$

Proof. Since $\kappa < 1$, $\mu > \mu^*$. Hence, solving [EP] using $\mu = \mu^*/\kappa$ produces an optimal solution to the original problem. The proof follows from Proposition 2. □

We infer from Propositions 5 and 6 that a numerical procedure allowed to control the accuracy of the solution can be devised to solve problems of the form [NLP]. Such a procedure can start with appropriately chosen μ and ϵ. The value of the penalty parameter μ can be increased according to some criteria, while the smoothing parameter ϵ can be decreased after each penalty minimization is completed. More precisely, if the solution of the penalty problem is ϵ-feasible, this is an indication that the penalty parameter μ is large enough. Therefore, if the smoothing parameter ϵ is below a certain accuracy tolerance, say $\epsilon_{\min}$, the algorithm terminates. In this case the statement of Proposition 6 provides an upper bound on the difference between the optimal value of the original problem and the optimal value of the penalty problem. If the solution of the penalty problem is not ϵ-feasible, it is understood that the penalty parameter μ is not large enough. Another round of penalty minimization is now carried out with a larger value of the penalty parameter. The smoothing parameter ϵ can be left unchanged in this case. We elaborate on the algorithm and the parameter update procedure in greater detail in §3. Before we do so, however, we state a final result that concerns the computation of a Lagrange multiplier vector in the context of penalty methods. The result requires the following assumption.

A3. The functions f_i for all $i = 0, \ldots, K$ are polyhedral convex, and the functions g_i for all $i = 1, \ldots, L$ are affine. Furthermore, the problem [NLP] has a unique Lagrange multiplier vector.

The convex conjugate function p_ϵ^* of p_ϵ is also used in the proof of the result below, and is given by

$$p_\epsilon^*(y_i) = \begin{cases} \frac{\epsilon}{2\mu} y_i^2 & \text{for } 0 \le y_i \le \mu, \\ \infty & \text{otherwise.} \end{cases} \tag{46}$$

PROPOSITION 7. *Under assumption* A3 *let* y^* *be the unique Lagrange multiplier vector of* [NLP]. *Let* $\tilde{x}$ *be an* ϵ*-feasible solution of* [SEP] *for some* ϵ. *Then* y^* *can be computed as*

$$y_i^* = \frac{dp_\epsilon}{dt}(\tilde{u}_i) = \max\left[0, \frac{\mu f_i(\tilde{x})}{\epsilon}\right] \quad \forall \ \ i = 1, \ldots, K, \tag{47}$$

provided that

$$\mu \ge y_i^* \quad \forall \ \ i = 1, \ldots, K$$

and

$$\epsilon \le \epsilon^*,$$

where ϵ^* *is a positive scalar.*

Proof. The proof is similar to the proof of Proposition 2 of [2]. Using the Fenchel duality theorem [22, Thm. 31.1], we have the following:

$$\inf_{x \in X} \left\{ f_0(x) + \sum_{i=1}^K p_\epsilon(f_i(x)) \right\} = \inf_u \left\{ q(u) + \sum_{i=1}^K p_\epsilon(u_i) \right\} = \max_y \left\{ g(y) - \sum_{i=1}^K p_\epsilon^*(y_i) \right\}, \tag{48}$$

where

$$q(u) = \inf_{x \in X} \{ f_0(x) | f_i(x) \le u \}$$

is the *perturbation function* corresponding to problem [NLP] and g is the *dual functional* defined as

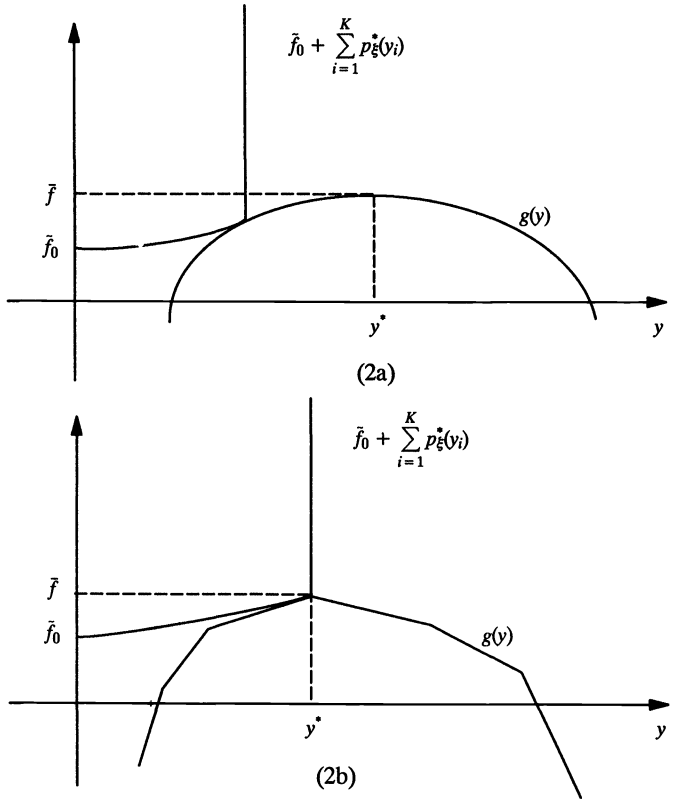
$$g(y) = \inf_u \left\{ q(u) + \sum_i u_i y_i \right\},$$

where $y \in \Re^K$. By Proposition 2 of [2], to obtain a Lagrange multiplier vector $y^* \in \Re^K$ of the original problem [NLP] via (47), it is necessary and sufficient that there exists a vector $\tilde{y} \in \Re^K$ such that

$$\mathbf{0} \in \partial g(\tilde{y}) \quad \text{and} \quad \mathbf{0} \in \partial \left(g - \sum_{i=1}^K p_\epsilon^* \right)(\tilde{y}). \tag{49}$$

But under assumption A3, $\epsilon^* \mathcal{B} \in \partial g(y^*)$ for some $\epsilon^* > 0$, where $\mathcal{B}$ is the unit ball in $\Re^K$. Thus, by the definition of the convex conjugate p_ϵ^*, y^* is a maximizer of g and thus a Lagrange multiplier vector, and a maximizer of $\partial(g - \sum_{i=1}^K p_\epsilon^*)(y^*)$ (i.e., y^* maximizes the right-hand side of (48)) provided that

$$\mu \ge y_i^* \quad \forall \ \ i = 1, \ldots, K$$



$$\inf_{x\in X}\left\{f_0(x) + \sum_{i=1}^{K} p_\xi(f_i(x))\right\} = \max_{y}\left\{g(y) - \sum_{i=1}^{K} p_\xi^*(y_i)\right\}$$

$$\tilde{f}_0 \overset{def}{=} \inf_{x\in X}\left\{f_0(x) + \sum_{i=1}^{K} p_\xi(f_i(x))\right\}$$

$$\bar{f} \overset{def}{=} \inf_{x\in X}\{f_0(x) | f_i(x) \leq 0 \forall\, i = 1, \ldots, K\}$$

FIG. 2. *Geometric interpretation of the linear-quadratic penalty function.*

and

$$\epsilon \leq \epsilon^*.$$

Thus, the result follows. □

We provide a visual aid in Fig. 2 to understand the ideas in Proposition 7. As can be observed from Fig. 2(a), a solution to the original problem cannot be expected by solving [SEP]. However, Fig. 2(b) shows that the threshold value for the penalty parameter needed to obtain a Lagrange multiplier vector to the original problem by solving the smooth penalty problem, can be characterized in terms of the magnitude of the Lagrange multiplier vector. Proposition 7 also provides the motivation for computing Lagrange multiplier estimates in the course of a penalty algorithm based on the linear-quadratic penalty function. These estimates can then be used to build an automatic update procedure for the penalty parameter μ. Such a procedure is developed in §3. A result similar to Proposition 7 is given in [2] using the quadratic

penalty function. However, the main difference is that we use a linear-quadratic penalty function with favorable approximate exactness properties, and we are able to recover the precise characterization of the threshold value for the penalty parameter μ.

3. Numerical examples. In this section we demonstrate a simple iterative scheme to solve problems of the form [NLP]. To start the LQP algorithm, an initial point $\tilde{x}_0$ is needed and initial values μ_0, ϵ_0 need to be specified. The following algorithmic framework was first suggested in [33] without any analysis and was applied to the solution of multicommodity network flows in [23] using parallel computing.

THE LQP ALGORITHM

Step 1. Using the violation $\tilde{x}_{k-1}$ as the starting point for evaluating the penalty function solve the problem

$$\min_{x \in X} \tilde{F}(x, \mu_k, \epsilon_k)$$

Let $\tilde{x}_k$ denote the optimal solution.

Step 2. If $\tilde{x}_k$ is ϵ-feasible and $\epsilon \leq \epsilon_{\min}$ terminate. Otherwise, update the penalty parameters μ and ϵ, set $k \leftarrow k+1$ and proceed from Step 1.

The parameter $\epsilon_{\min} > 0$ is a final infeasibility tolerance and in this study its value is 10^{-3}. At Step 2, there are two situations that may arise:

1. The point $\tilde{x}_k$ is ϵ-feasible but $\epsilon > \epsilon_{\min}$. In this case leave μ unchanged and decrease ϵ as

$$\epsilon_{k+1} = \eta_1 \max \tilde{u}_i$$

where $\tilde{u}_i = f_i(\tilde{x}_k)$ for all $i = 1, \ldots, K$ and $\eta_1 = 0.5$.

2. The point $\tilde{x}_k$ is not ϵ-feasible which indicates that the value of the penalty parameter is not large enough. Increase μ as

$$\mu_{k+1} = \eta_2 \mu_k$$

where $\eta_2 = 2.0$ and keep the current value ϵ_k.

By virtue of Propositions 5 and 6 solving the penalty problem using a value of the penalty parameter μ larger than the threshold, we can obtain a solution that is ϵ-feasible. This statement is true for any $\epsilon > 0$, in particular for $\epsilon = \epsilon_{\min}$. Since the penalty parameter is adjusted only a finite number of times to attain the threshold in Step 2 of the iterative scheme, we can obtain a solution that is $\epsilon_{\min}$-feasible in a finite number of penalty minimizations. The solution also provides an error bound from the optimal value of the original problem as stated in Proposition 6.

Since p_ϵ is continuous in x and X is a compact subset of $\Re^n$, then there exists, by the Weierstrass theorem, a finite point that minimizes $\tilde{F}(x, \mu_k, \epsilon_k)$ for every μ_k and ϵ_k. This guarantees the existence of the iterates $\tilde{x}_k$ at Step 1 of the LQP algorithm. The above observations can now be summarized in the following proposition.

PROPOSITION 8. *Starting from $x_0 \in X$ and given μ_0, ϵ_0 and a final tolerance $\epsilon_{\min} > 0$ such that $\epsilon_0 \geq \epsilon_{\min}$, the iterative scheme produces an $\epsilon_{\min}$-feasible solution in a finite number of executions of Step* 1.

In our numerical experiments with the network structured problems we found the largest objective function coefficient to be a good choice for the initial value of the penalty parameter μ. An alternative update procedure for μ can also be built based on the following observations.

sing the optimality conditions for [NLP] from §2.2, we have

$$\nabla f_0(x^*) = -\sum_{i=1}^{K} y_i^* \nabla f_i(x^*) - \sum_{i=1}^{L} z_i^* \nabla g_i(x^*), \tag{50}$$

$$y_i^* f_i(x^*) = 0 \quad \forall \quad i = 1, \ldots, K, \qquad z_i^* g_i(x^*) = 0 \quad \forall \quad i = 1, \ldots, L, \tag{51}$$

$$y_i^* \geq 0 \quad \forall \quad i = 1, \ldots, K, \qquad z_i^* \geq 0 \quad \forall \quad i = 1, \ldots, L, \tag{52}$$

$$f_i(x^*) \leq 0 \quad \forall \quad i = 1, \ldots, K, \qquad g_i(x^*) \leq 0 \quad \forall \quad i = 1, \ldots, L. \tag{53}$$

Let $\tilde{x}$ be an optimal solution for [SEP], and $\tilde{z}$ be a Lagrange multiplier vector. Using the optimality conditions from §2.2, we have

$$\nabla f_0(\tilde{x}) = -\sum_{i \in \mathcal{A}} \frac{\mu f_i^+(\tilde{x})}{\epsilon} \nabla f_i(\tilde{x}) - \sum_{i \in \mathcal{V}} \mu \nabla f_i(\tilde{x}) - \sum_{i=1}^{L} \tilde{z}_i \nabla g_i(\tilde{x}), \tag{54}$$

$$\tilde{z}_i g_i(\tilde{x}) = 0 \quad \forall \quad i = 1, \ldots, L, \tag{55}$$

$$\tilde{z}_i \geq 0 \quad \forall \quad i = 1, \ldots, L, \tag{56}$$

$$g_i(\tilde{x}) \leq 0 \quad \forall \quad i = 1, \ldots, L. \tag{57}$$

Let

$$(y_\epsilon)_i = \begin{cases} \mu & \text{for } i \in \mathcal{V}, \\ \frac{\mu f_i(\tilde{x})}{\epsilon} & \text{for } i \in \mathcal{A}, \\ 0 & \text{otherwise.} \end{cases} \tag{58}$$

Notice that y_ϵ is obtained as the first derivative of the penalty function as stated in (47). Condition (54) reduces to

$$\nabla f_0(\tilde{x}) = -\sum_{i} y_\epsilon \nabla f_i(\tilde{x}) - \sum_{i=1}^{K} \tilde{z}_i \nabla g_i(\tilde{x}),$$

which is reminiscent of the condition (50) for optimality in the original problem. Thus a solution $\tilde{x}$ to the smooth penalty problem and the multipliers $(y_\epsilon, \tilde{z})$ defined above, satisfy conditions (50)-(53) of the conditions for optimality of the original problem, with the exception of primal feasibility and complementary slackness for inequalities $f_i(x) \leq 0$. Optimality for the problem [NLP] would be assured if we had primal feasibility (i.e., $f_i(\tilde{x}) \leq 0$; (53)) and complementary slackness (i.e., $y_{\epsilon_i} f_i(\tilde{x}) = 0$; (51)).

In solving [SEP] we are content to achieve an $\tilde{x}$ that satisfies (54)–(57); that is ϵ-feasible, i.e., $\mathcal{V} = \emptyset$ (and hence $f_i(\tilde{x}) \leq \epsilon$) and that (almost) satisfies complementary slackness (51). Since, by construction $y_\epsilon \geq 0$, and

$$f_i(\tilde{x}) \leq \epsilon \quad \forall \quad i = 1, \ldots, K \quad (\epsilon - \text{feasibility})$$

by ϵ-feasibility of $\tilde{x}$, this implies (by definition of y_ϵ)

$$y_\epsilon \mathbf{f}(\tilde{x}) \leq \mu\epsilon \quad (\epsilon - \text{complementary slackness}),$$

where $\mathbf{f} : \Re^n \mapsto \Re^K$. The above analysis gives insight as to how two of the key parameters in the LQP algorithm can be adjusted. First, ϵ should start out relatively large and should be reduced so that $\mu\epsilon$ becomes an acceptable complementary slackness error in the limit. If no constraint is violated (i.e., $\mathcal{V} = \emptyset$) then we leave μ unchanged. If at least one constraint is violated, we increase μ according to the formula

$$\mu \leftarrow \max_i \left\{ \frac{\mu f_i(\tilde{x})}{\epsilon} \right\}.$$

The initial value ϵ_0 for the smoothing parameter was taken as the largest amount of violation in the penalized constraints evaluated at the initial solution. Either one of the above update procedures can be used in the implementation of the algorithm. Experimentation shows that the choice of update procedure does not seem to impair the progress of the algorithm.

A software system GENOS/LP was built to solve nonlinear programs with embedded network structures. GENOS is a collection of FORTRAN subroutines that contain implementations of the network simplex algorithm, the truncated Newton algorithm, and the simplicial decomposition algorithm, to solve network flow problems; see [21]. This program was extended in GENOS/LP for the solution of embedded networks.

We conclude this section with two numerical examples illustrating the performance of the smooth penalty method. Iterations refer to the number of smooth penalty minimizations. Both examples were solved using GENOS/LP. The purpose of including low-dimensional numerical examples is to illustrate the potential of the penalty method in delivering reasonably accurate solutions to linear programming problems.

Example 1. The first example is taken from [1]. It is a multicommodity network flow problem with two commodities.

$$\begin{aligned} \max & 100x_1 + 120x_2 + 90x_3 + 80x_4 + 70x_5 + 140x_6 \\ & +40x_7 + 20x_8 + 30x_9 + 20x_{10} + 40x_{11} + 10x_{12} \end{aligned}$$

s.t.

$$\begin{aligned} x_1 + x_2 + x_3 &= 25, \\ x_4 + x_5 + x_6 &= 15, \\ x_1 + x_4 &= 20, \\ x_2 + x_5 &= 10, \\ x_3 + x_6 &= 10, \\ x_7 + x_8 + x_9 &= 50, \\ x_{10} + x_{11} + x_{12} &= 30, \\ x_7 + x_{10} &= 20, \\ x_8 + x_{11} &= 40, \\ x_9 + x_{12} &= 20, \end{aligned}$$

$$x_1 + x_7 \leq 30,$$
$$x_3 + x_9 \leq 30,$$
$$0 \leq x_i \leq 75, \qquad i = 1, \ldots, 12.$$

The problem was solved using the smooth penalty function after placing the nonnetwork inequalities

$$x_1 + x_7 \leq 30\,, \qquad x_3 + x_9 \leq 30 \tag{59}$$

in the objective function. The solution and the optimal value reported by the smooth penalty method and the Lagrange multiplier vector from (39) matched exactly the values reported by MINOS, when the problem was solved as a linear program. Solution statistics are given in Table 1. Each line gives the statistics at the beginning of a penalty minimization. The values of penalty parameter μ, and the smoothing parameter ϵ, and the value of the original objective function are reported along with the absolute degree of infeasibility in the penalized constraints. The first line corresponds to the solution obtained by ignoring the nonnetwork constraints, i.e., by solving the network flow problem for each commodity.

TABLE 1
Penalty method statistics for Example 1.

Iterations	μ	ϵ	$\max_i \tilde{u}_i$	Objective value
1	140	5.0	5.0	7300
2	280	0.714	1.428	7157
3	560	5.1×10^{-2}	0.10	7104
4	1120	1.82×10^{-3}	3.64×10^{-3}	7100.14
5	1120	1.82×10^{-3}	7×10^{-4}	7100

Example 2. This example is taken from [13].

$$\min 10x_2 + 2x_3 + x_4 + 3x_5 + 4x_6$$
s.t.

$$x_1 + x_2 = 10,$$
$$-x_1 + x_3 + x_4 - x_5 = 0,$$
$$-x_2 - x_3 + x_5 + x_6 = 70,$$
$$-x_4 - x_6 = -10,$$
$$10x_1 - 2x_3 + 3x_4 - 2x_5 \leq 16,$$
$$x_1 + 4x_3 + x_5 \leq 10,$$
$$0 \leq x_1 \leq 12,$$
$$0 \leq x_2 \leq 18,$$
$$0 \leq x_3 \leq 5,$$
$$0 \leq x_4 \leq 12,$$
$$0 \leq x_5 \leq 1,$$
$$0 \leq x_6 \leq 16.$$

The problem has an embedded network structure with two nonnetwork constraints (the first two inequalities). Those were placed in the objective function thus obtaining

a nonlinear program with network flow constraints. To obtain a starting point, the problem is solved ignoring the nonnetwork constraints. Solution statistics are reported in Table 2. The same problem was solved as a linear program using MINOS 5.0. The MINOS optimal solution is 124.00.

TABLE 2
Solution statistics for Example 2.

Iterations	μ	ϵ	$\max_i \tilde{u}_i$	Objective value
1	10	114.00	114.00	10.00
2	20	5.70	11.41	112.58
3	40	0.14	0.28	123.71
4	80	1.75×10^{-3}	3.50×10^{-3}	123.99
5	80	1.75×10^{-3}	2×10^{-4}	123.99

The optimal solution vector reported by MINOS is $x^* = (1.230, 8.769, 0, 1.230, 0, 8.769)$. The optimal solution given by the penalty method is $\tilde{x} = (1.231, 8.769, 0, 1.231, 0, 8.769)$. The optimal value of component of the Lagrange multiplier vector corresponding to the inequality

$$10x_1 - 2x_3 + 3x_4 - 2x_5 \leq 16$$

is 1.00 as reported by MINOS. The Lagrange multiplier estimate given by the penalty method is 1.11. These examples illustrate that the smooth penalty method may provide reasonably accurate solutions to the original problem.

4. Application to network-structured problems. A motivation for using the linear-quadratic penalty function is to solve large-scale optimization problems characterized by some special structure. In this section we address problems with embedded network structures of the following form:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & f(x) \\ \text{subject to} \quad & Ax = b, \\ & Sx \leq d, \\ & 0 \leq x \leq u, \end{aligned}$$

where:

- f: $\Re^n \to \Re$ is the objective function, assumed to be convex and continuously differentiable;
- x $\in \Re^n$ is the vector of decision variables which represent flows on a graph;
- A is an $m \times n$ constraint matrix with network structure. It could be the node-arc incidence matrix of a network flow problem, or a block-diagonal matrix where each block is a node-arc incidence matrix as occurs in multicommodity network flows, stochastic networks, and time-staged problems;
- S is the $s \times n$ matrix of side (i.e., nonnetwork) constraints imposed on the network flow variables;
- u $\in \Re^n$ are upper bounds on the flow variables x;
- $b \in \Re^m, d \in \Re^s$ are the right-hand side coefficients of the constraints.

As we pointed out in the introduction, a penalty function may be used to place the nonnetwork constraints into the objective function. This produces a nonlinear network program at Step 1 of the iterative scheme. The nonlinear network program is then solved using a linearization technique such as simplicial decomposition [20]. This approach was used in [33] to solve multicommodity network flow problems and for parallel computations in [23]. In both cases the results were very encouraging.

In §4.1, we report on applications of the linear-quadratic penalty technique to the solution of network flow problems with general side constraints as opposed to the more structured multicommodity problems. We summarize our experience with the solution of (i) large-scale personnel management problems from the Naval Personnel Research Center, and (ii) nonlinear matrix estimation problems from World Bank econometric studies.

4.1. Solving the naval personnel assignment problem. In this section we report numerical results obtained using GENOS/LP on two naval personnel assignment problems. These models determine the allocation of the Navy enlisted personnel to a fleet of combat units and to mission areas within these units. We experimented with linear and nonlinear versions of the two test problems. The results with the linear problems also appear in the conference proceedings [24]. The results with the nonlinear problems are new.

The first model, NAVY, is a simplified version of the complete model HUGE-NAVY. The size and characteristics of both problems are given in Table 3.

TABLE 3
Characteristics of naval personnel assignment problem.

Problem	Linear Programming		Network		Number of side constraints
	Rows	Columns	Nodes	Arcs	
NAVY	4144	6842	3457	6841	687
HUGENAVY	36013	64542	30639	64541	5374

Both problems were solved on a CRAY Y-MP. Table 4 summarizes the solution statistics of the LQP algorithm. All times are in CPU seconds exclusive of input/output. It is interesting to note that the larger Navy problem is solved in a time that is very close to the solution time of the smaller problem. This can be attributed to the larger number of iterations taken by the algorithm in the case of problem NAVY because the smaller problem is more tightly constrained than the larger. Since the iterates generated by the LQP algorithm only become feasible on termination, the previous observation leads to the conclusion that, though much larger in size, HUGENAVY is a relatively easier problem to solve for our method.

We also report the solution time by MINOS for NAVY. The LQP algorithm was outperformed by MINOS on this problem. However, MINOS was not able to produce a feasible solution to HUGENAVY after three hours of CPU time on a CRAY Y–MP whereas this problem was solved within five minutes using the LQP method. On the other hand, the same problem was solved in less than three minutes using the OB1 code based on interior point methods. This indicates that the LQP algorithm based on nonlinear programming technology is competitive with current linear programming technology. We also note that the optimal objective function value was computed to five digits of accuracy by the LQP method as confirmed for both problems by the OB1. It was also communicated to us that the NETSID code of Kennington and Wissman [14] for linear network flow problems with side constraints based on primal partitioning was not able to solve the linear versions of the Navy problems [15].

TABLE 4
Performance of the LQP *algorithm on the naval problems.*

Problem	GENOS/LP time	MINOS time	OB1 time
NAVY (CRAY Y-MP)	194	NA	NA
NAVY (DEC 5100)	1560	600	NA
HUGENAVY(CRAY Y-MP)	276	> 3 hrs.	150

TABLE 5
Performance of the LQP *algorithm on the nonlinear naval personnel assignment problems.*

Problem	Total time
NAVYQ	456
HUGENAVYQ	1137

We also experimented with nonlinear versions of NAVY and HUGENAVY problems. We refer to these problems as NAVYQ and HUGENAVYQ, where the objective function is a separable quadratic function of the form $\sum_j a_j x_j^2$. For the problem NAVYQ, the coefficients a_j are precisely the coefficients given in the linear model. For the HUGENAVYQ problem the coefficient vector was taken to be identically unity. We report the solution statistics in Table 5. Both results were obtained on a CRAY Y-MP.

With both problems, the infeasibility tolerance $\epsilon_{\min} = 10^{-5}$ was attained on termination.

We did not run the nonlinear problems with MINOS due to the very large computing time.

4.2. Solving nonlinear matrix estimation problems. In this section we report results with side constrained versions of two matrix estimation problems from the World Bank. The matrix estimation problem consists of adjusting the entries of social accounting matrices (SAM) for an economy and can be formulated as a nonlinear network optimization problem, see [32]. The first problem SAMKE is a SAM model for Kenya and the second problem SAMBOT is a SAM model for Botswana. Both problems were derived from econometric studies conducted at the World Bank. Both problems have separable objective functions. The problem SAMKE has a weighted entropy objective function of the form $ax \cdot (\log \frac{x}{b} - 1.0)$ and the problem SAMBOT has a quadratic objective function of the form $a \cdot (x - b)^2$ for any flow variable x.

Side constraints have the following form:

$$\sum_{(ij)\in\mathcal{E}} x_{ij} \geq \beta \sum_{(ij)\in\mathcal{E}} x_{ij}^*,$$

where $\mathcal{E}$ is an arbitrary subset of the arcs of the underlying graph and x_{ij} is the flow variable for arc (i,j), x_{ij}^* are the optimal flows obtained by solving the original matrix estimation problem without side constraints and $\beta \in (0,1]$. Characteristics of the network component of the problems are summarized in Table 6.

Both problems were solved with an increasing number of side constraints using GENOS/LP and MINOS. Tests were performed on a DECstation 3100. All times are given in CPU seconds. In Fig. 3 we provide the variation of the CPU time taken by GENOS/LP and MINOS on SAMBOT as a function of the number of side constraints added to the problem. MINOS is the only software package that we can compare our results against for these problems. We observe that while MINOS time does not

TABLE 6
Characteristics of matrix estimation problems.

Problem	Network	
	Nodes	Arcs
SAMKE	50	202
SAMBOT	128	662

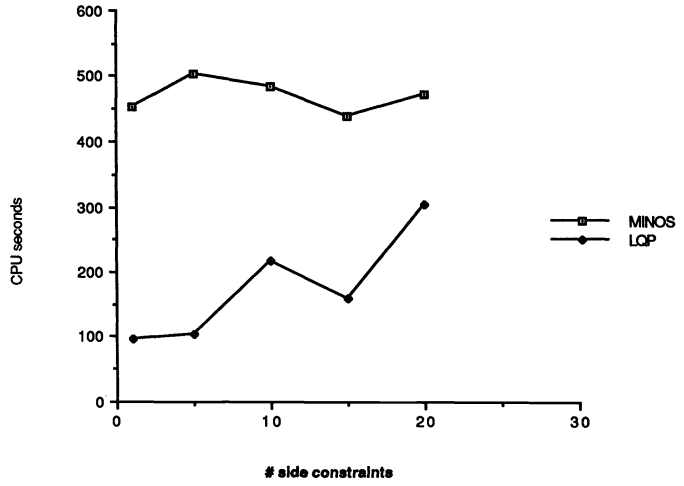


FIG. 3. *Solution time for constrained matrix estimation problem* SAMBOT *as a function of the number of side constraints with* GENOS/LP *and* MINOS.

increase considerably, GENOS/LP outperforms MINOS by a significant margin for a small number of side constraints. The advantage of GENOS/LP is reduced as the number of side constraints increases. In Fig. 4, we plot the optimal values reported by GENOS/LP and MINOS with the problem SAMBOT as a function of the number of side constraints. In all experiments, GENOS/LP produced reasonably accurate solutions to the problem.

5. Application to nonlinear minimax problems. In this section we develop an application of the quadratic smoothing technique to nonlinear minimax problems. This illustrates the potential of the penalty method for solving a class of problems different than the multicommodity flows in [29] and [33] or the embedded network structures of the previous section. We also point out that computational experience in the literature on minimax optimization is usually limited to small problems, e.g., [5], [9], [13]. The minimax problem is stated as follows:

$$\min_{x \in \Re^n} \max_{i \in \mathcal{M}} f_i(x), \tag{60}$$

where the $f_i : \Re^n \mapsto \Re$ are continuously differentiable functions and $\mathcal{M} = \{1, \ldots, m\}$. Equivalently the minimax problem can be stated as the following nonlinear program:

$$\text{[MINMAX]} \qquad \underset{x,\, z}{\text{minimize}} \quad z$$

$$\text{subject to} \quad f_i(x) \leq z \quad \forall \quad i = 1, \ldots, m.$$

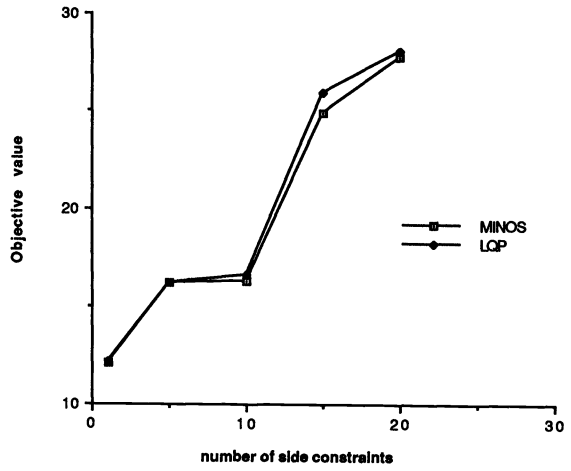


FIG. 4. *The optimal value for constrained matrix estimation problem* SAMBOT *as a function of the number of side constraints with* GENOS/LP *and* MINOS.

We propose to compute an approximate solution to problem [MINMAX] by solving the following problem:

$$\text{(61)} \qquad \text{[MINSEP]} \qquad \text{minimize} \quad \left[z + \sum_{i=1}^{m} p_\epsilon(f_i(x) - z) \right].$$

The motivation for applying the LQP algorithm to minimax problems lies in the following observation. In the previous sections we established that the threshold value of the penalty parameter μ to achieve ϵ-feasibility is a function of the maximum component of a Lagrange multiplier vector to the original problem. The components of a Lagrange multiplier vector in problem [MINMAX] are in the interval $[0, 1]$. (This fact is readily verified by studying the optimality conditions for [MINMAX].) As a result of this observation in an iterative scheme where the solution of problem [MINMAX] is approximately obtained by solving a finite number of penalty problems, one can start with a value of μ in the range $(0, 1]$. This property enables one to alleviate the problems associated with the ill conditioning of the penalty problem with very large values of μ. An iterative scheme can be devised based on the steps described in §3. In this case the penalty minimization step, Step 1, is an unconstrained problem. To start the algorithm, an initial point $(\tilde{x}_0, z_0)$ is needed and initial values μ_0, ϵ_0 for the penalty parameter μ and the smoothing parameter ϵ need to be specified. A natural choice for the smoothing parameter ϵ is to take ϵ_0 as a fraction of the function values at x_0, i.e., $\epsilon_0 = \beta \max_i(f_i^+(\tilde{x}_0))$ with $\beta \in (0, 1]$.

The effectiveness of the iterative scheme is contingent upon the success of the unconstrained minimizations in Step 1. We use a Newton method to solve the penalty problem [15] that is available as a subroutine in the IMSL Library [12]. Although the penalty function p_ϵ is not twice continuously differentiable everywhere, the Newton algorithm from [12] performed satisfactorily on the test problems we tried. This point deserves further investigation.

A parametrized test problem is used to illustrate the performance of the penalty algorithm on nonlinear minmax problems. The problem is an unconstrained version

of [24, Prob. 8]. It is given as

$$\min_x \max_{i=1\in\mathcal{M}} f_i(x),$$

with

$$\begin{aligned} f_1(x) &= x_1^2 + x_2 + x_3 + \cdots + x_n - 1, \\ f_2(x) &= x_1 + x_2^2 + x_3 + \cdots + x_n - 1, \\ f_3(x) &= x_1 + 2x_2^2 + x_3 + \cdots + x_n - 1, \\ &\vdots \end{aligned}$$

$$\left.\begin{aligned} f_i(x) &= x_1 + x_2 + x_3 + \cdots + x_{\frac{i+2}{2}}^2 + \cdots + x_n - 1 \\ f_{i+1}(x) &= x_1 + x_2 + x_3 + \cdots + 2x_{\frac{i+2}{2}}^2 + \cdots + x_n - 1 \end{aligned}\right\} i \text{ even},$$

$$f_m(x) = x_1 + x_2 + x_3 + \cdots + x_n^2 - 1,$$

where $m = 2n - 2$, n even. We have run the example with $n = 4, 10, 20, 30, 40, 60, 70,$ and 80.

The computation was done on a VAX 6400 mainframe computer running VMS. Solution statistics are reported in Table 7 in the following format: The first three columns correspond to the size of the problem, the initial value of the penalty parameter μ and the final infeasibility tolerance respectively. The remaining columns report the number of penalty minimizations (MIT), the total number of inner iterations of the Newton algorithm (NIT), the total number of function (NF), gradient (NG) and Hessian (NH) evaluations and the solution reported on completion.

We also solved these problems using two state-of-the-art nonlinear programming software packages, MINOS 5.1 and the recently released code Lancelot of Conn, Gould, and Toint [7]. In Lancelot experiments we report the number of function and gradient evaluations that were required to reach a specified level of accuracy. Both Lancelot and MINOS were run with default parameters. The LQP algorithm was also run without any tuning of its parameters. The level of accuracy in Lancelot was controlled using the parameter **gradient_tolerance**. Its value is set to 10^{-3} in our experiments. We remark that the solution reported by Lancelot is slightly more accurate than the solution reported by the penalty method with respect to the MINOS solution. The statistics are reported in Table 7. The column corresponding to MINOS gives the total number of calls to the function and gradient evaluations subroutine. The starting point in all runs was taken to be $x_i = 10$, for all $i = 1, 2, 3 \ldots, n - 1$, and $x_n = 0$. This is the starting point used in [18].

On the average we find that we have four digits of accuracy in the optimal value reported by the penalty algorithm with respect to the MINOS solution. The LQP method requires substantially fewer function and gradient evaluations than MINOS or Lancelot. We do not report solution times as these experiments were carried out on different machines.

3. Conclusions. In this paper we considered a smoothing approximation to the ℓ_1 exact penalty function and investigated its properties. The resulting smooth

TABLE 7
Solution statistics for the minimax using the LQP, MINOS, *and Lancelot.*

(n,m)	μ_0	ϵ_{fin}	LQP					MINOS 5.1	Lancelot	
			MIT	NIT	NF	NG	NH	NF/NG	NF	NG
(4,6)	1	10^{-2}	4	33	59	37	33	NA	50	44
(10,18)	0.5	10^{-2}	4	43	81	47	43	96	196	166
(20,38)	0.7	10^{-2}	3	27	54	30	27	175	288	247
(30,58)	0.7	10^{-2}	3	43	69	46	43	249	288	249
(40,78)	0.7	10^{-2}	3	39	81	42	39	NA	362	349
(60,118)	0.7	10^{-2}	3	40	82	43	40	NA	465	418
(70,138)	0.6	10^{-2}	3	41	98	44	41	NA	533	478
(80,158)	0.7	10^{-2}	3	25	68	28	25	NA	656	581

MIT = Number of penalty minimizations,
NIT = Number of Newton iterations,
NF = Number of function evaluations,
NG = Number of gradient evaluations,
NH = Number of Hessian evaluations,
NA = not available.

penalty function inherits some interesting approximate exactness properties from its nondifferentiable counterpart. Furthermore, computational experiences indicate that it has the potential of becoming a powerful tool for solving convex constrained optimization problems. It was used with success to solve large network flow problems with side constraints and nonlinear minimax problems.

Acknowledgments. We would like to thank R. Dembo, M. Guignard-Spielberg, P. Harker, and S. Wright for numerous insightful suggestions and comments on an earlier draft. The comments of two anonymous referees led to substantial improvements of this paper. Kristjan Jonasson kindly performed the Lancelot experiments.

REFERENCES

[1] S. P. BRADLEY, A. C. HAX, AND T. L. MAGNANTI, *Applied Mathematical Programming*, Addison-Wesley, Redwood City, CA, 1977.
[2] D. P. BERTSEKAS, *Necessary and sufficient conditions for a penalty method to be exact*, Math. Programming, 9 (1975), pp. 87–99.
[3] ———, *Non-differentiable optimization via approximation*, Math. Programming Study, 3 (1973), pp. 1–25.
[4] ———, *Constrained Optimization and Lagrange Multiplier Methods*, Academic Press, New York, 1982.
[5] C. CHARALAMBOUS, *A lower bound for the controlling parameter of the exact penalty functions*, Math. Programming, 15 (1978), pp. 278–290.
[6] C. CHARALAMBOUS AND A. R. CONN, *An efficient method to solve the minimax problem directly*, SIAM J. Numer. Anal., 15 (1978), pp. 162–187.
[7] A. R. CONN, N. I. M. GOULD, AND PH. L. TOINT, *Lancelot: A Fortran Package for Large-Scale Nonlinear Optimization (Release A)*, Springer-Verlag, Heidelberg, Berlin, New York, 1992.
[8] A. R. CONN AND Y. LI, *A structure exploiting algorithm for nonlinear minimax problems*, SIAM J. Optim., 2 (1992), pp. 242–263.
[9] G. DI PILLO AND L. GRIPPO, *An exact penalty function method with global convergence properties for nonlinear programming problems*, Math. Programming, 36 (1986), pp. 1–18.
[10] D. M. GAY, *Algorithm* 611: *subroutine for unconstrained optimization using a model trust region approach*, ACM Trans. Math. Software, 9 (1983), pp. 503–524.
[11] P. J. HUBER, *Robust Statistics*, John Wiley & Sons, New York, 1981.
[12] IMSL User's Guide, IMSL, Houston, Texas, 1987.
[13] J. L. KENNINGTON AND R. V. HELGASON, *Algorithms for Network Programming*, Wiley Interscience, New York, 1980.

[14] J. L. KENNINGTON AND A. WISSMAN, *NETSID user's guide: a Fortran program for network flow problems with side constraints*, Tech. Report 86–OR–01, Department of Operations Research, Southern Methodist University, Dallas, TX, 1986.

[15] I. KRASS, *Private communication*, Navy Personnel Research and Development Center, San Diego, CA, 1991.

[16] I. J. LUSTIG, R. MARSTEN, AND D. SHANNO, *Computational experience with a primal-dual interior point method for linear programming*, Linear Algebra Appl., 152 (1991), pp. 191–222.

[17] K. MADSEN AND H. B. NIELSEN, *A finite smoothing algorithm for ℓ_1 estimation*, SIAM J. Optim., 3 (1993), pp. 223–235.

[18] K. MADSEN AND H. SCHJÆR-JACOBSEN, *Linearly constrained minimax optimization*, Math. Programming, 14 (1978), pp. 208–223.

[19] J. M. MULVEY, R. VANDERBEIM, AND S. A. ZENIOS, *Robust Optimization: Modelling Framework and Computations*, Report 91–06–04, Decision Sciences Department, University of Pennsylvania, Philadelphia, PA, 1991.

[20] J. M. MULVEY, S. A. ZENIOS, AND D. P. AHLFELD, *Simplicial decomposition for convex generalized networks*, J. Information and Optim. Sci., 11 (1990), pp. 359–387.

[21] J. M. MULVEY AND S. A. ZENIOS, *GENOS: A generalized network optimization system user's guide*, Report 87–12–12, Decision Sciences Department, University of Pennsylvania, Philadelphia PA, 1987.

[22] B. A. MURTAGH AND M. A. SAUNDERS, *MINOS* 5.1 *user's guide*, Report SOL 83–20R, December 1983, revised January 1987, Stanford University, Stanford, CA, 1987.

[23] M. C. PINAR AND S. A. ZENIOS, *Parallel decomposition of multicommodity flows using smooth penalty functions*, ORSA J. Comput., 4 (1992), pp. 235–249.

[24] M. C. PINAR AND S. A. ZENIOS, *Naval personnel assignment: An application of linear-quadratic penalty functions*, in Computer Science and Operations Research: New Developments in Their Interfaces, O. Balcı, R. Sharda, and S.A. Zenios, eds., Pergamon Press, Elmsford, NY, pp. 43–56, 1992.

[25] R. T. ROCKAFELLAR, *Convex Analysis*, Princeton University Press, Princeton, NJ, 1970.

[26] ———, *Linear-quadratic programming and optimal control*, SIAM J. Control Optim., 25 (1987), pp. 781–814.

[27] ———, *Computational schemes for large-scale problems in extended linear-quadratic programming*, Math. Programming, 48 (1990), pp. 447–474.

[28] R. T. ROCKAFELLAR AND R. J.-B. WETS, *Scenarios and policy aggregation in optimization under uncertainty*, Math. Oper. Res., 16 (1991), pp. 1–29.

[29] G. L. SCHULTZ AND R. R. MEYER, *An interior point method for block angular optimization*, SIAM J. Optim., 1 (1991), pp. 583–602.

[30] K. TRUEMPER, *Note on finite convergence of exterior penalty functions*, Management Sci., 21 (1975), pp. 600–606.

[31] I. ZANG, *A smoothing-out technique for min-max optimization*, Math. Programming, 19 (1980), pp. 61–77.

[32] S. A. ZENIOS, A. DRUD, AND J. M. MULVEY, *Balancing large social accounting matrices with nonlinear network programming*, Networks, 19 (1989), pp. 569–585.

[33] S. A. ZENIOS, M. Ç. PINAR, AND R. S. DEMBO, *A smooth penalty function algorithm for network structured problems*, Report 90-12-05, Decision Sciences Department, University of Pennsylvania, Philadelphia PA, 1990; European J. Oper. Res, to appear.

SIAM J. OPTIMIZATION
Vol. 4, No. 3, pp. 512–520, August 1994
©1994 Society for Industrial and Applied Mathematics
004

A COMPLEXITY ANALYSIS FOR INTERIOR-POINT ALGORITHMS BASED ON KARMARKAR'S POTENTIAL FUNCTION*

JUN JI† AND YINYU YE‡

Abstract. A new complexity analysis for constant potential reduction algorithms for linear programming is considered. Using Karmarkar's primal-based potential function, it is shown that for a class of linear programs the number of iterations required is at most $O(mt + n\log(n))$, where m is the number of equations and n is the number of variables in the standard linear programming form, and t can be interpreted as the number of significant figures required in the final solution. It is also shown that this result holds in expectation for some randomly generated problems. At the same time, it is shown that Karmarkar's algorithm requires at least $\Omega(n)$ iterations to solve Anstreicher's linear program, if the initial solution is poor but valid.

Key words. linear programming, interior-point algorithm, potential function

AMS subject classification. 90C05

1. Potential function and linear programming.

Consider the linear programming (LP) problem:

$$\text{(P)} \qquad \begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax = b, \qquad x \geq 0, \end{array}$$

where $c \in \mathbf{R}^n$, $A \in \mathbf{R}^{m\times n}$, and $b \in \mathbf{R}^m$ are given, $x \in \mathbf{R}^n$, and ^{T} denotes the transpose. The dual to (P) can be written as

$$\text{(D)} \qquad \begin{array}{ll} \text{maximize} & b^T y \\ \text{subject to} & s = c - A^T y \geq 0, \end{array}$$

where vectors $y \in \mathbf{R}^m$ and $s \in \mathbf{R}^n$. The components of s are called dual slacks. We assume that the optimal objective value of (P) and (D) exists and we denote it by z^*.

Karmarkar's [10] original potential function is:

$$\phi(x, z^*) = \rho \ln(c^T x - z^*) - \sum_{j=1}^{n} \ln(x_j),$$

which is a primal-based potential function. This function was used with $\rho = n$ in Karmarkar's projective potential reduction algorithm for solving linear programs in the canonical form:

$$\text{(KP)} \qquad \begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax = 0,\ e^T x = n \text{ and } x \geq 0, \end{array}$$

*Received by the editors April 12, 1990; accepted for publication (in revised form) April 25, 1993.

†Department of Mathematics, University of Iowa, Iowa City, Iowa 52242. Current address, Department of Mathematics and Computer Science, Valdosta State University, Valdosta, Georgia 31698 (junji@grits.valdosta.peachnet.edu).

‡Department of Management Sciences, University of Iowa, Iowa City, Iowa 52242 (bbuyin@vaxa.weeg.uiowa.edu). Research supported in part by National Science Foundation grant DDM-8922636 and the Iowa Business School Summer Grant.

where $z^* = 0$ and e is the vector of all ones. It was also used with $\rho = n + 1$ in the projective algorithm for solving linear programs of the form (P) with unknown z^* (e.g., Anstreicher [3], Gay [6], de Ghellinck and Vial [7], Gonzaga [9], Steger [13], Todd and Burrell [16], and Ye and Kojima [20]). It was later used with $\rho \geq n + \sqrt{n}$ to develop an affine potential reduction algorithm (e.g., Freund [5], Gonzaga [8], and Ye [18]).

The primal-dual potential function

$$\psi(x, s) = \rho \ln(x^T s) - \sum_{j=1}^{n} \ln(x_j s_j)$$

with $n < \rho < \infty$ was first introduced by Tenabe [14] and Todd and Ye [17]. Then it was used to develop a primal-dual, affine or projective, potential reduction algorithm in [18] and [19] (see also Freund [5]).

Complexity analysis for each of the various potential reduction algorithms is based on showing that on each iteration the potential function can be reduced by a constant. The work on each iteration, $O(n^3)$ operations, is dominated by the computation of an orthogonal projection. The total number of iterations required to terminate the algorithm is determined by a convergence (or stopping) criterion that is different for each of the different potential functions.

The primal potential function, under appropriate assumptions, requires its potential value to lie below $-O(\rho t)$, i.e.,

$$\phi(x^k, z^*) \leq -O(\rho t)$$

to reach

$$\text{(A)} \qquad c^T x^k - z^* \leq 2^{-t}.$$

Thus, most of the primal-based potential reduction algorithms require $O(\rho t)$ iterations to terminate the iterative process.

On the other hand, the primal-dual potential function can be written in an equivalent form

$$\psi(x, s) = (\rho - n) \ln(x^T s) - \sum_{j=1}^{n} \ln\left(\frac{x_j s_j}{x^T s}\right).$$

From the inequality of the geometric mean and the arithmetic mean, we have

$$-\sum_{j=1}^{n} \ln\left(\frac{x_j s_j}{x^T s}\right) \geq n \ln n.$$

Hence,

$$(\rho - n) \ln(c^T x - b^T y) = (\rho - n) \ln(x^T s) \leq \psi(x, s) - n \ln n \leq \psi(x, s),$$

which gives the exact amount, $-\Omega((\rho - n)t)$, by which $\psi(\bullet, \bullet)$ should be reduced to reach

$$c^T x^k - b^T y^k \leq 2^{-t}.$$

This property guarantees that the primal-dual potential reduction algorithm can be terminated in $O((\rho - n)t)$ iterations if $\psi(x^0, s^0)$ is bounded above by $O((\rho - n)t)$ and $\psi(\bullet, \bullet)$ is reduced by a constant amount on each iteration.

Thus, for $\rho = n + \sqrt{n}$ the primal algorithm needs $O(nt)$ iterations, while the primal-dual algorithm needs only $O(\sqrt{n}t)$. In the complexity analysis the use of the primal-dual potential function improves the convergence result significantly over what was found with the primal-based potential function.

Primal-based potential reduction algorithms would require a potential reduction of $\Omega(\sqrt{n})$ to compare favorably with primal-dual potential reduction algorithms. However, several "worst-case" examples show that the primal potential function is reduced at most by a constant in each iteration through the entire iterative process (see Anstreicher [1], [2] and McDiarmid [11]). The question is: Can the complexity result of primal-based potential reduction algorithms, in particular Karmarkar's algorithm, be improved even though the reduction in $\phi(\bullet, \bullet)$ is only $\Omega(1)$ per iteration?

In this paper, using a different convergence analysis, we give an improved complexity result for primal-based potential reduction algorithms applied to certain linear programs. In fact, if the primal-based potential function is used to measure solution progress, we can show that the complexity result for some of the "worst-case" examples is still quite good. We also show, however, that the algorithm requires at least $\Omega(n)$ iterations to solve Anstreicher's linear program for a poor choice of the initial solution. Furthermore, for a class of linear programs, we use this convergence criterion to prove that the number of iterations required to terminate the algorithm is $O(mt + n\log(n))$. Termination criterion (A) deals with the absolute error. In practice, the relative error

$$\text{(B)} \qquad \frac{c^T x^k - z^*}{c^T x^0 - z^*} \leq 2^{-t}$$

is a more interesting measure of algorithm performance, and is used as the termination criterion throughout the paper.

2. A convergence result for the primal potential function. In this section, we use $\rho = n$ in the potential function for linear programming problems of the form (KP). Given any dual *optimal* solution y^* and the corresponding slack vector $s^* \geq 0$, the primal potential function can be written as

$$\phi(x, z^*) = n\ln(x^T s^*) - \sum_{j=1}^{n} \ln(x_j).$$

For simplicity we let all zero-components in s^* be the first $\bar{m}$ components. Note that $\bar{m} = m$ if s^* is dual nondegenerate. Then

$$\begin{aligned}\phi(x, z^*) &= \bar{m}\log(x^T s^*) + (n - \bar{m})\log(x^T s^*) - \sum_{j=1}^{\bar{m}} \log(x_j) - \sum_{j=\bar{m}+1}^{n} \log(x_j) \\ &= \bar{m}\log(x^T s^*) + \eta(x, s^*) + \sum_{j=\bar{m}+1}^{n} \log(s_j^*) - \sum_{j=1}^{\bar{m}} \log(x_j),\end{aligned}$$

where

$$\eta(x, s^*) = (n - \bar{m})\log(x^T s^*) - \sum_{j=\bar{m}+1}^{n} \log(x_j s_j^*).$$

Again using the inequality of the geometric mean and the arithmetic mean, we have

$$\eta(x, s^*) \geq (n - \bar{m})\log(n - \bar{m}).$$

Note that $\eta(x, s^*)$ can be viewed as a measure of how poorly x is "centered" with respect to s^*. Now we have

$$\phi(x^k, z^*) - \phi(x^0, z^*) = \bar{m} \log\left(\frac{(x^k)^T s^*}{(x^0)^T s^*}\right) + \eta(x^k, s^*) - \eta(x^0, s^*) - \sum_{j=1}^{\bar{m}} \log\left(\frac{x_j^k}{x_j^0}\right)$$

or

$$(1)\quad \bar{m} \log\left(\frac{(x^k)^T s^*}{(x^0)^T s^*}\right) = \phi(x^k, z^*) - \phi(x^0, z^*) + \eta(x^0, s^*) - \eta(x^k, s^*) + \sum_{j=1}^{\bar{m}} \log\left(\frac{x_j^k}{x_j^0}\right).$$

Let us look at the right-hand side of the equality (1). The difference of the first two terms $\phi(x^k, z^*) - \phi(x^0, z^*)$ represents the total net reduction of the potential function; the third and fourth terms measure how centered x^0 and x^k, respectively, are with respect to s^*, and the last term is determined by the ratios of x_j^k/x_j^0 for $j = 1, 2, \ldots, \bar{m}$. Note that in (KP) if $x^0 = e$, then $\sum_{j=1}^{\bar{m}} \log(x_j^k/x_j^0)$ is bounded above by $\bar{m} \log(n/\bar{m})$. Thus, to attain

$$\bar{m} \log\left(\frac{(x^k)^T s^*}{(x^0)^T s^*}\right) < -\Omega(\bar{m}t),$$

as required to satisfy the termination criterion (B), it suffices to have

$$(2)\qquad \phi(x^k, z^*) - \phi(x^0, z^*) \leq -\Omega(\bar{m}t) - \bar{m} \log(n/\bar{m}) - (\eta(x^0, s^*) - \eta(x^k, s^*)).$$

We state this observation as the following theorem.

THEOREM 1. *Consider an* LP *problem of the form* (KP), *with starting feasible solution* $x^0 = e$. *Then for* x^k *to meet the termination criterion* (B), *it suffices that* x^k *satisfies the condition* (2).

Note that the total required reduction of the potential function from (2) depends only on $\bar{m}$, t, $\log(n)$ and $\eta(x^0, s^*) - \eta(x^k, s^*)$. The reason for using $\eta(\bullet, s^*)$ in Theorem 1 will become clear later. For some special cases, we immediately obtain the following improved convergence result.

COROLLARY 1. *To solve* (KP) *with the initial feasible solution* $x^0 = e$, *if* $\eta(x^0, s^*) - \eta(x^k, s^*) \leq O(m \log(n))$ *and* $\bar{m} = O(m)$, *then the total number of iterations required to terminate the algorithm is* $O(mt + m \log(n))$; *if* $\eta(x^0, s^*) - \eta(x^k, s^*) \leq O(n \log(n))$ *and* $\bar{m} = O(m)$, *then the total number of iterations required to terminate the algorithm is* $O(mt + n \log(n))$.

Clearly, $\eta(x^0, s^*) - \eta(x^k, s^*)$ plays a key role in our analysis. Again, $-\eta(x^k, s^*)$ is bounded above by $-(n - \bar{m}) \log(n - \bar{m}) \leq 0$. Therefore, the number of iterations to terminate the algorithm depends only on $\bar{m}$, t, $\log(n)$ and the *initial* $\eta(x^0, s^*)$. If somehow we can evaluate or control the term $\eta(x^0, s^*) - \eta(x^k, s^*)$, then we may derive better complexity results. This is what we explore in the next section.

3. An application of the convergence result. In this section, we demonstrate how our convergence criterion discussed above may be applied to some problems and examples proposed to analyze the performance of interior point algorithms. One frequently used problem is

$$(\text{SP})\qquad \begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & e^T x = n \text{ and } x \geq 0, \end{array}$$

where $z^* = 0$. This is the simplest problem in Karmarkar canonical form. Sonnevend, Stoer, and Zhao [12] showed that the path-following algorithm solves this type of problem in $O(n^{1/4}t)$ iterations, which is considered a "good" case in their complexity analysis. On the other hand, Anstreicher [1] illustrated that the primal potential function, even with perfect line search, is reduced only by a constant at each iteration when

$$c = (0, 1, \ldots, 1, \gamma, \beta)^T$$

for some positive constants γ and β (also see McDiarmid [11]). This example is considered a "bad" case for the projective algorithm. However, using Theorem 1 and Corollary 1, we can immediately obtain the following complexity result for (SP).

COROLLARY 2. *Let $\bar{m}$, the number of zero-components of c, be $O(1)$, and assume that the remaining positive components of c have $|\log(c_j)| \leq O(\log(n))$, $j = \bar{m}+1, \ldots, n$. Then, primal-based potential reduction algorithms solve* (SP) *in $O(t + n\log(n))$ iterations.*

Proof. In this problem, we can take $s^* = c$ and $x^0 = e$. Obviously, x^0 is feasible for (SP) and $\eta(x^0, s^*)$ is controlled by

$$\eta(x^0, s^*) = (n - \bar{m})\log(c^T e) - \sum_{j=\bar{m}+1}^{n} \log(c_j) \leq O((n - \bar{m})\log(n)).$$

Thus,

$$\eta(x^0, s^*) - \eta(x^k, s^*) \leq O((n - \bar{m})\log(n)) - (n - \bar{m})\log(n - \bar{m}) \leq O(n\log(n)).$$

Therefore, the proof follows Corollary 1. □

Now consider Anstreicher's problem when it is solved by Karmarkar's algorithm with $x^0 = e$. Since the relation (1) is an *equality*, we can use it to provide a complexity lower bound for the algorithm. During the iterative process, an interesting property of Anstreicher's problem is

$$\eta(x^k, s^*) = \eta(x^0, s^*) = \eta(e, s^*) \quad \text{for all} \quad k \geq 0$$

for a pair of (γ, β) with

$$\beta > \gamma > 0.525 \quad \text{and} \quad \gamma + \beta = \sqrt{3}.$$

This is because either

$$X^k s^* = X^k c = \delta^k (0, 1, \ldots, 1, \gamma, \beta)^T$$

or

$$X^k s^* = X^k c = \delta^k (0, 1, \ldots, 1, \beta, \gamma)^T,$$

where δ^k is a positive scaler [1], and $\eta(x^k, s^*)$ is invariant under scaling and permuting. Thus, the equality (1) can be written as

$$\log\left(\frac{(x^k)^T s^*}{(x^0)^T s^*}\right) = \phi(x^k, z^*) - \phi(x^0, z^*) + \log\left(\frac{x_1^k}{x_1^0}\right). \tag{3}$$

Also note that $\bar{m} = 1$ here. Therefore, we are able to simplify the derivation of Anstreicher's result.

COROLLARY 3 (Anstreicher). *Starting at $x^0 = e$, Karmarkar's algorithm requires $\Theta(t + \log(n))$ iterations to obtain an x^k satisfying the termination criterion* (B).

Proof. We only need to evaluate x_1^k when the algorithm reaches the criterion (B), i.e., when

$$\frac{c^T x^k}{c^T e} \leq 2^{-t}$$

or

$$\sum_{j=2}^{n-2} x_j^k + \gamma x_{n-1}^k + \beta x_n^k = c^T x^k \leq 2^{-t} c^T e = 2^{-t}(n + \sqrt{3} - 3).$$

However, this relation implies

$$\gamma \left(\sum_{j=2}^{n} x_j^k \right) \leq 2^{-t}(n + \sqrt{3} - 3),$$

which together with the simplex constraint in (KP) further implies

$$x_1^k \geq n - \frac{2^{-t}(n + \sqrt{3} - 3)}{\gamma} > n(1 - 2^{-t}/\gamma).$$

Thus,

$$\log x_1^k \geq \log(n) + \log(1 - 2^{-t}/\gamma),$$

which establishes the desired result. □

Note that we have obtained the above result merely using equality (1) or (3), without deriving a rather complex and analytical formula for x^k at every iteration. Our analysis demonstrates that the primal potential function may still be a useful tool in analyzing algorithm performance. (Of course, the old stopping criterion is a poor one.) Also note that the complexity result for (SP) is much better than the "best" result analyzed by Sonnevend, Stoer, and Zhao in a path-following algorithm [12]. Moreover, the above result indicates that the convergence rate is not superlinear (this has been discussed in Bayer and Lagarias [4]).

To see how important the choice of an initial solution can be, consider the following. Let

$$x_1^0 = 2^{-n} \quad \text{and} \quad x_j^0 = \frac{n - 2^{-n}}{n - 1} \quad \text{for} \quad j = 2, 3, \ldots, n$$

in Anstreicher's problem. In this case, x^0 is a point close to a (wrong) face of the feasible region. Then the initial potential value will be

$$\phi(x^0, z^*) \leq O(n \log(n)),$$

which indicates that although x^0 is a poor starting point for the projective algorithm, it is nonetheless a valid one (i.e., $\phi(x^0, z^*) \leq O(nL)$ in rational models [10], where L is the data size). Note that the algorithm needs, with t fixed, a total net reduction of at least $\Omega(n) = \log(n/x_1^0)$ to reach the termination criteria (B) due to equality (3). Furthermore, a total net reduction of $\Omega(n)$ requires $\Omega(n)$ iterations in the problem. This example illustrates that for a fixed t, the use of a poor starting point can increase the required number of iterations to $\Omega(n)$, compared to only $O(\log(n))$ using the starting point e.

4. Probabilistic analysis of $\eta(e, s^*)$. As we mentioned before, $\eta(x^0, s^*) - \eta(x^k, s^*)$ plays a key role in our complexity analysis. Since $-\eta(x^k, s^*)$ is always bounded above by $-(n-\bar{m})\log(n-\bar{m}) \leq 0$, we would like to examine the bound for

$$\eta(x^0, s^*) = \eta(e, s^*) = (n-\bar{m})\log\left(\sum_{j=\bar{m}+1}^{n} s_j^*\right) - \sum_{j=\bar{m}+1}^{n}\log(s_j^*),$$

when s^* is randomly generated. Some random problems are proposed in Todd [15], where the positive components in an s^* are independently and identically drawn from the Gaussian distribution $|N(0,1)|$. Using Todd's model, for example, we may construct random (KP) problems as follows: Generate any matrix $(B,\ N) \in R^{m\times(n-2)}$, and generate $\hat{x}_B$ and $\hat{s}_N$ where their components are independently and identically drawn from the distribution $|N(0,1)|$. Let $b = B\hat{x}_B$, and in the (KP) form let

$$A = (-b,\ B,\ N,\ b - Be - Ne) \quad \text{and} \quad c = (0,\ 0,\ \hat{s}_N,\ 0)^T.$$

Then, $z^* = 0$ and a feasible $x^0 = e$ for this problem. Furthermore, with probability one, an optimal primal solution is

$$x_1^* = \frac{n}{1+e^T\hat{x}_B} > 0,\ x_B^* = \frac{n\hat{x}_B}{1+e^T\hat{x}_B} > 0 \quad \text{and} \quad x_j^* = 0, \quad j = m+2,\ldots,n$$

and an optimal dual slack vector is $s^* = c$, where $s_N^* = \hat{s}_N > 0$.

We prove the following lemma for the function $g(\bullet) : R_+^d \to R$

$$g(\xi) = d\log\left(\sum_{j=1}^{d}\xi_j\right) - \sum_{j=1}^{d}\log(\xi_j).$$

LEMMA 1. *Consider a probability distribution function on the sample space $\mathcal{E} = (0,\infty)$ with $0 \leq p(x) \leq 1$ as the probability density function, and let the ξ_j's be independently drawn from this probability distribution. Then*

$$d\log(d) \leq E(g(.)) \leq d\log(d) + d(1+\log(c)),$$

where

$$c = \int_{\mathcal{E}} xp(x)dx.$$

Proof. $E(g(.)) \geq d\log(d)$ comes directly from the geometric-arithmetic mean inequality. Note that

$$\begin{aligned}\int_{\mathcal{E}}\log(x)p(x)dx &= \int_{(0,1]}\log(x)p(x)dx + \int_{(1,\infty)}\log(x)p(x)dx\\ &\geq \int_{(0,1]}\log(x)p(x)dx\\ &\geq \int_{(0,1]}\log(x)dx\\ &\geq \int_0^1\log(x)dx = -1.\end{aligned}$$

On the other hand,

$$E(\log(e^T\xi)) \leq \log(E(e^T\xi)) = \log(dc).$$

Overall, we have the desired result. □

Since $g(\xi)$ is invariant under scaling and any random variable ξ can be scaled such that the density function of the scaled variable is less than one in the sample space, the assumption $0 \leq p(x) \leq 1$ in the lemma is not restrictive at all. For example, for the uniform distribution on $(0, 1)$ we have $c = \frac{1}{2}$; for the exponential distribution with parameter $\lambda = 1$ we have $c = 1$; for the Gaussian distribution $|N(0, 1)|$, $c = 2/\sqrt{2\pi}$. From Lemma 1, we see that the expected value of $\eta(x^0, s^*) - \eta(x^k, s^*)$ is bounded above by $O(n)$ if s^* is generated as above. Thus, from Theorem 1 we have the following theorem.

THEOREM 2. *Given a problem of the form* (KP), *with A and c randomly generated as above, and starting at the feasible solution* $x^0 = e$, *the expected number of iterations for primal-based potential reduction algorithms to generate a feasible solution satisfying the termination criterion* (B) *is bounded above by* $O(\bar{m}t + n)$.

5. Further remarks. The above complexity analysis can be incorporated in any monotonic primal potential reduction algorithm with lower bound ($z^k \leq z^*$) updating scheme, since

$$\phi(x^{k+1}, z^*) - \phi(x^k, z^*) \leq \phi(x^{k+1}, z^k) - \phi(x^k, z^k)$$

as long as $c^T x^{k+1} \leq c^T x^k$ (see for example, Anstreicher [3] and Todd and Burrell [16]). In addition, the analysis can easily be extended to algorithms that directly solve a standard form problem (P) (see, for example, [20]), rather than the specialized problem (KP).

REFERENCES

[1] K. M. ANSTREICHER, *On the performance of Karmarkar's algorithm over a sequence of iterations*, SIAM J. Optim., 1 (1991), pp. 22–29.

[2] ———, *A worst-case step in Karmarkar's algorithm*, Math. Oper. Res., 14 (1989), pp. 294–302.

[3] ———, *A monotonic projective algorithm for fractional linear programming*, Algorithmica, 1 (1986), pp. 483–498.

[4] D. A. BAYER AND J. C. LAGARIAS, *Karmarkar's linear programming algorithm and Newton's method*, Math. Programming, 50 (1991), pp. 291–330.

[5] R. M. FREUND, *Polynomial-time algorithms for linear programming based only on primal scaling and projected gradients of a potential function*, Math. Programming, 51 (1991), pp. 203–222.

[6] D. M. GAY, *A variant of Karmarkar's linear programming algorithm for problems in standard form*, Math. Programming, 37 (1987), pp. 81–90.

[7] G. DE GHELLINCK AND J.-P. VIAL, *A polynomial Newton method for linear programming*, Algorithmica, 1 (1986), pp. 425–454.

[8] C. C. GONZAGA, *Polynomial affine algorithms for linear programming*, Math. Programming, 49 (1991), pp. 7–22.

[9] ———, *Conical projection algorithms for linear programming*, Math. Programming, 43 (1989), pp. 151–173.

[10] N. KARMARKAR, *A new polynomial-time algorithm for linear programming*, Combinatorica, 4 (1984), pp. 373–395.

[11] C. MCDIARMID, *On the improvement per iteration in Karmarkar's method for linear programming*, Math. Programming, 46 (1990), pp. 299–320.

[12] G. SONNEVEND, J. STOER, AND G. ZHAO, *On the complexity of following the central path of linear programs by linear extrapolation* II, Math. Programming, 52 (1991), pp. 527–553.

[13] A. STEGER, *An extension of Karmarkar's algorithm for bounded linear programming problems*, M.S. thesis, State University of New York at Stonybrook, 1985.

[14] K. TENABE, *Complementarity-enforcing centered Newton method for mathematical programming*, in New Methods for Linear Programming, K. Tone, ed., The Institute of Statistical Mathematics, Tokyo, Japan, 1987, pp. 118–144.

[15] M. J. TODD, *Probabilistic models for linear programming*, Math. Oper. Res., 16 (1991), pp. 671–693.

[16] M. J. TODD AND B. P. BURRELL, *An extension of Karmarkar's algorithm for linear programming using dual variables*, Algorithmica, 1 (1986), pp. 409–424.

[17] M. J. TODD AND Y. YE, *A centered projective algorithm for linear programming*, Math. Oper. Res., 15 (1990), pp. 508–529.

[18] Y. YE, *An $O(n^3L)$ potential reduction algorithm for linear programming*, Math. Programming, 50 (1991), pp. 239–258.

[19] ———, *A class of projective transformations for linear programming*, SIAM J. Comput., 19 (1990), pp. 457–466.

[20] Y. YE AND M. KOJIMA, *Recovering optimal dual solutions in Karmarkar's polynomial algorithm for linear programming*, Math. Programming, 39 (1987), pp. 305–317.

SIAM J. OPTIMIZATION
Vol. 4, No. 3, pp. 521–536, August 1994
©1994 Society for Industrial and Applied Mathematics
005

PROBLEMS OF HIERARCHICAL OPTIMIZATION IN FINITE DIMENSIONS*

RUOXIN ZHANG†

Abstract. In this paper two-level hierarchical optimization problems in finite dimensions are formally defined and the existence of solutions as well as various necessary optimality conditions are studied under nonsmooth settings. The assumptions made to the problems allow the objectives of both levels to be nondifferentiable and the solutions to the lower-level problems to be multivalued. Techniques of nonsmooth analysis are the primary tools for deriving the results.

Key words. hierarchical optimization, nonsmooth analysis, upper-Lipschitz multifunctions, functions with strong minimum, semi-Lipschitz conditions.

AMS subject classifications. 90C30, 49J52, 90D65

1. Introduction.

Problems involving optimization on two distinct levels, one of which is dominant over the other, are called *two-level hierarchical optimization* problems. At the lower level is an ordinary optimization problem in elements $y \in \mathbb{R}^m$ and parameterized by elements $x \in \mathbb{R}^n$:

$$(Q(x)) \qquad \text{minimize } g(x,y) \text{ over } y \in \mathbb{R}^m \text{ subject to } y \in \Omega(x),$$

where g, called the lower-level objective function, is defined on $\mathbb{R}^n \times \mathbb{R}^m$ (possibly extended-real-valued) and Ω is a multifunction from $\mathbb{R}^n$ to $\mathbb{R}^m$ called the *constraint multifunction.* At the upper level is an ordinary optimization problem with respect to x, which however depends in its formulation on the potential outcomes in solving the lower-level problem.

The hierarchical problems can be classified according to whether the upper-level problem depends on the lower-level problem primarily through the optimal value function $s(x) = \min_{y\in\Omega(x)} g(x,y)$ or through the optimal solution multifunction

$$S(x) = \operatorname*{argmin}_{y\in\Omega(x)} g(x,y).$$

Problems of the first kind typically take the form

$$\text{minimize } \tilde{f}(x, s(x)) \text{ over } x \in \mathbb{R}^n \text{ subject to } G(x, s(x)) \leq 0,$$

where $\tilde{f}$ is a function on $\mathbb{R}^n \times \mathbb{R}$ and G is a mapping from $\mathbb{R}^n \times \mathbb{R}$ to some other space $\mathbb{R}^k$. Problems of the second kind instead take the form

$$\text{minimize } f(x,y) \text{ over } (x,y) \in D \text{ subject to } y \in R(x),$$

where D is a closed subset in $\mathbb{R}^n \times \mathbb{R}^m$, f, called the upper-level objective function, is defined on $\mathbb{R}^n \times \mathbb{R}^m$ and $R : \mathbb{R}^n \rightrightarrows \mathbb{R}^m$ is the *rational response multifunction* satisfying $R(x) \subset S(x)$ for all x. Generally, we refer to the problems of the second

*Received by the editors February 24, 1992; accepted for publication April 20, 1993.

†Department of Mathematics and Computer Science, Northern Michigan University, Marquette, Michigan 49855-5340 (farz%nmumus@cunyvm.cuny.edu).

kind as *two-level problems* because they actually include the problems of the first kind through a certain reformulation of the latter.

Two main choices for the rational response multifunction R in the two-level problems are

$$R(x) = \operatorname*{argmin}_{y \in S(x)} f(x,y) \qquad \text{and} \qquad R(x) = \operatorname*{argmax}_{y \in S(x)} f(x,y),$$

and the corresponding problems are referred to as being *cooperative* and *noncooperative*, respectively. In the cooperative case, the agent responsible for solving the upper-level problem (the leader) is able to induce the agent or agents responsible for the lower-level problem (the follower) to accept whichever solution to the lower-level problem is most convenient for the upper-level problem. In the noncooperative case, the upper-level agent has no such influence and proceeds according to the worst that may happen.

The cooperative case of the basic two-level problem can be written equivalently as

$$(P_C) \qquad \text{minimize} \quad f(x,y) \ \text{ over } \ (x,y) \in D \ \text{ subject to } \ y \in S(x),$$

and is the primary concern of this paper. Its distinguishing characteristic, compared to other categories of optimization, is thus seen in the presence of a constraint on (x,y) in terms of an optimality condition with respect to a subproblem, rather than in the traditional terms of inequalities or equalities. This is a major complication requiring techniques of analysis different from those that have served elsewhere.

Two-level problems, due to their hierarchical structures, have been recognized as fruitful mathematical models for economic applications where two agents have different interests (they may or may not conflict) with a biased information pattern. The application examples may be found in [4], [8], [24], and references therein.

Two-level hierarchical optimization problems have been studied by a number of authors including, for instance, Bard [4] who derived the optimality conditions for problems with differentiable objectives and an algorithm for linear-quadratic problems; Loridan and Morgan [13] who dealt with so-called *approximate solutions*; Ishizuka and Aiyoshi [11] for solution algorithms; and Lucchetti, Mignanego, and Pieri [14] who considered existence and optimality conditions for problems with single-valued lower-level solution and some other special cases with polyhedral Ω. For a complete survey on linear two-level problems, see Wen and Shu [28]. In this paper, we derive existence and optimality conditions for problems where the objective functions are allowed to be nondifferentiable, the solution of the lower-level problem is multivalued, and the constraint multifunction Ω has fewer restrictions compared to existing results. Techniques developed in nonsmooth analysis are the primary tools later in the paper.

Our main focus is on the cooperative two-level problem P_C and therefore, unless specified, the "two-level hierarchical problem" refers to the cooperative case only without confusion. The organization of the materials is as follows.

The existence of solutions for problems that may have unbounded lower-level solution sets is studied in §1. In §2 we study the possible selections of *exact penalty* functions to the lower-level problem to ease the constraints into objective. Then, in §3, optimality conditions are derived based on the lower-level solution multifunction S. In §5, where the lower-level problem is assumed to be convex, we combine the results of §§3 and 4 to obtain some optimality conditions for the two-level problems of a rather general type. The optimality conditions obtained are applicable to the piecewise linear-quadratic two-level problems under certain Lipschitz assumptions on the objectives of both levels.

2. Existence of solutions to the hierarchical cooperative problems. In this section, we shall derive some results on the existence of solutions to problem (P_C). In fact, if we denote the graph of the multifunction S by $\operatorname{gph} S$ and the indicator function of $\operatorname{gph} S$ by $\delta_{\operatorname{gph} S}$, our primary concern is on the closedness of the set $\operatorname{gph} S$ since the existence of a solution to P_C is implied by the inf-compactness of the function $f+\delta_{\operatorname{gph} S}$ meaning that $f+\delta_{\operatorname{gph} S}$ is lower-semicontinuous, proper, and for every $\rho \in \mathbb{R}$ the level-set $\operatorname{lev}_\rho(f+\delta_{\operatorname{gph} S}) := \{(x,y) \mid f(x,y) \le \rho,\ y \in S(x)\}$ is bounded.

The closedness of $\operatorname{gph} S$ has been established by Hogan [9, Thm. 8] for continuous multifunction Ω and continuous g. For more general cases where the multifunction $x \mapsto \{y \mid g(x,y) \le \alpha, (x,y) \in \operatorname{gph} S\}$ is locally bounded for some α, Rockafellar has obtained the closedness of $\operatorname{gph} S$ as well as the compactness of $S(x)$ under other conditions [21, Thm. 3.1]. In what follows we will deal with problems in which S may be unbounded at some points in $\operatorname{dom}\Omega$ where $\operatorname{dom}\Omega := \{x \mid \Omega(x) \neq \emptyset\}$.

Let $p(x)$ be the optimal value to the lower-level problem $(Q(x))$. Then $p(x) := \inf\{g(x,y) | y \in \Omega(x)\}$. For any $\rho > 0$ define a set $E_\rho \subset \mathbb{R}^n$ by

$$E_\rho := \{x \mid \exists y \in \Omega(x) \text{ with } |y| \le \rho, g(x,y) < \infty\},$$

and a function $p_\rho : \mathbb{R}^n \to \overline{\mathbb{R}}$ by

$$p_\rho(x) := \inf\{g(x,y) \mid y \in \Omega(x) \cap \rho\mathbb{B}_m\},$$

where $\mathbb{B}_m$ stands for the unit ball in $\mathbb{R}^m$.

PROPOSITION 2.1. *If g is lower-semicontinuous and proper, and for every $\rho > 0$ the set E_ρ is closed and $p(x)$ is upper-semicontinuous on E_ρ, then S has closed graph.*

Proof. For any $x^\nu \to \bar{x}$ and $y^\nu \to \bar{y}$ with $y^\nu \in S(x^\nu)$, there exists a $\rho > 0$ such that $y^\nu \in \rho\mathbb{B}$ and $x^\nu \in E_\rho$ for sufficient large ν, therefore $\bar{x} \in E_\rho$. Since $p(x)$ is upper-semicontinuous on E_ρ and $p(x^\nu) = g(x^\nu, y^\nu)$, one has

$$p(\bar{x}) \ge \limsup_{\nu\to\infty} p(x^\nu) \ge \liminf_{\nu\to\infty} g(x^\nu, y^\nu).$$

By the lower-semicontinuity of g, it follows that $p(\bar{x}) \ge g(\bar{x}, \bar{y})$ and therefore $(\bar{x}, \bar{y}) \in \operatorname{gph} S$. □

Here we give a simple example where the assumptions in Proposition 2.1 are satisfied. Let

$$g(x,y) = \begin{cases} 0 & \text{if } y \ge 1/|x|, \\ 1 & \text{otherwise,} \end{cases}$$

and $\Omega(x) = \mathbb{R}^m$. Then $E_\rho = \mathbb{R}^n$ for $\rho > 0$ and

$$p(x) = \begin{cases} 1 & \text{if } x = 0, \\ 0 & \text{otherwise.} \end{cases}$$

The set $\operatorname{gph} S = \{(x,y) \mid y \ge 1/|x|\}$ is clearly closed.

COROLLARY 2.2. *If g is lower-semicontinuous and finite, Ω has closed graph and for every $x \in E := \cup_{\rho>0} E_\rho$ there exists an $N_x > 0$ such that $p_\rho(x)$ is continuous at x for all $\rho \ge N_x$. Then S has closed graph.*

Proof. For every fixed $x \in E$, notice that $p_\rho(x)$ is nonincreasing in ρ and $p(x) = \inf\{p_\rho(x) \mid \rho > 0\}$. Therefore there exists an N_x such that

$$\limsup_{x' \xrightarrow{E} x} p(x') \le \limsup_{x' \xrightarrow{E} x} p_\rho(x') = p_\rho(x) \quad \forall \rho \ge N_x.$$

Set $\rho \to \infty$, we have

$$\limsup_{x' \xrightarrow{E} x} p(x') \le p(x).$$

It follows that p is upper-semicontinuous on E and E_ρ. To show the closedness of E_ρ, let $\rho > 0$ and $x^\nu \in E_\rho$ with $x^\nu \to \bar{x}$. Then there exists y^ν satisfying $|y^\nu| \le \rho$ and $g(x^\nu, y^\nu) < \infty$. Since $\{y^\nu\}$ is bounded, by passing to the subsequences, we have $y^\nu \to \bar{y}$ with $|\bar{y}| < \infty$ and $g(\bar{x}, \bar{y}) \le \liminf g(x^\nu, y^\nu) < \infty$. We have $\bar{x} \in E_\rho$. Therefore the claim follows from Proposition 2.1. □

If in the previous example we modify g to

$$g(x,y) = \begin{cases} 0 & \text{if } y \ge 1/|x|, \quad x \ge 0, \\ 1 & \text{otherwise,} \end{cases}$$

remaining $\Omega(x) = \mathbb{R}^m$, then $p(x)$ is not upper-semicontinuous and $\operatorname{gph} S$ is no longer closed. An example that indicates that the conditions in Proposition 2.1 are not sufficient is obtained when $\Omega(x) = \mathbb{R}^m$ and

$$g(x,y) = \begin{cases} 0 & \text{if } |x| \le 1, \quad y \ge 0, \\ \infty & \text{otherwise.} \end{cases}$$

In this case $\operatorname{gph} S$ is closed even though p is not upper-semicontinuous on E. Notice that we have adopted the convention that $\operatorname{argmin}\{\infty\} = \emptyset$. If we let $g(x,y) = 1$ in place of ∞, then $\operatorname{gph} S = \{(x,y) \mid |x| \le 1 \text{ or } y < 0\}$ which is no longer closed.

PROPOSITION 2.3. *In addition to the assumptions in Proposition* 2.1, *if f is inf-compact and there exists $(x,y) \in \operatorname{gph} S$ with $f(x,y) < \infty$, then the solution to the problem (P_G) exists.*

Proof. This is a direct consequence of Proposition 2.1. □

3. Penalty representations of lower-level problems. We start this section with a definition on the *semi-Lipschitz continuity* of a function h defined on $\mathbb{R}^k$ relative to a subset $C \subset \mathbb{R}^k$.

DEFINITION 3.1. *The function h is said to satisfy a semi-Lipschitz condition relative to C if there exists a constant (modulus) $L \ge 0$ such that*

$$h(x) - h(x') \le L|x - x'| \quad \forall\, x \in C, \quad x' \notin C,$$

and it is said to satisfy a local semi-Lipschitz condition relative to C if there exists $L \ge 0$ and $\varepsilon > 0$ such that

$$h(x) - h(x') \le L|x - x'| \quad \forall\, x \in C,\ x' \in C + \varepsilon\mathbb{B}, \quad x' \notin C.$$

A function satisfying a semi-Lipschitz condition relative to C need not be Lipschitz continuous on C although a (globally) Lipschitz continuous function always satisfies a semi-Lipschitz condition with respect to any set. In fact, an arbitrary function satisfies a semi-Lipschitz condition relative to any of its level sets. We denote the Euclidean distance from $y \in \mathbb{R}^m$ to the set $\Omega(x)$ by $\operatorname{dist}(y|\Omega(x))$ for any fixed $x \in \mathbb{R}^n$ and write $d_\Omega(x,y) := \operatorname{dist}(y|\Omega(x))$.

LEMMA 3.2. *Suppose $\Omega(x)$ is closed and $g(x,\cdot)$ satisfies a semi-Lipschitz condition relative to the set $\Omega(x)$ with modulus $L(x)$ for each $x \in \operatorname{dom}\Omega$. If for any $x \in \operatorname{dom}\Omega$, there exists $\beta(x) > L(x)$ such that $\bar{y}$ satisfies*

$$g(x,\bar{y}) + \beta(x) d_\Omega(x,\bar{y}) \le g(x,y) \quad \forall\, y \in \Omega(x), \tag{3.1}$$

then $\bar{y} \in \Omega(x)$.

Proof. If $\bar{y} \notin \Omega(x)$, from (3.1) and the semi-Lipschitz condition of $g(x,\cdot)$, one would have

$$\beta(x)d_\Omega(x,\bar{y}) \le g(x,y) - g(x,\bar{y}) \le L(x)|y-\bar{y}| \quad \forall y \in \Omega(x).$$

Taking the infimum on the right side over $y \in \Omega(x)$ yields $\beta(x)d_\Omega(x,\bar{y}) \le L(x)d_\Omega(x,\bar{y})$. Since $L(x) < \beta(x)$, there would be a contradiction if $d_\Omega(x,\bar{y}) \ne 0$. Therefore $\bar{y} \in \Omega(x)$. □

PROPOSITION 3.3. *Suppose* $\Omega(x)$ *is closed and* $g(x,\cdot)$ *satisfies a semi-Lipschitz condition relative to the set* $\Omega(x)$ *with modulus* $L(x)$ *for each* $x \in \operatorname{dom}\Omega$. *Let* $\pi : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}_+$ *and* $\beta : \mathbb{R}^n \to \mathbb{R}_+$ *be functions satisfying* $\beta(x) > L(x)$ *for every* $x \in \operatorname{dom}\Omega$ *and*

$$\pi(x,y) \begin{cases} \ge \beta(x)d_\Omega(x,y) & \text{if } y \notin \Omega(x), \\ = 0 & \text{otherwise.} \end{cases} \tag{3.2}$$

Then for any $x \in \operatorname{dom}\Omega$, *one has* $S(x) = \operatorname{argmin}\{g(x,y) + \pi(x,y) \mid y \in \mathbb{R}^m\}$.

Proof. For any $\bar{y} \in \operatorname{argmin}\{g(x,y) + \pi(x,y) \mid y \in \mathbb{R}^m\}$, we have

$$g(x,\bar{y}) + \beta(x)d_\Omega(x,\bar{y}) \le g(x,\bar{y}) + \pi(x,\bar{y}) \le g(x,y) + \pi(x,y) \quad \forall y \in \mathbb{R}^m.$$

In particular, $g(x,\bar{y}) + \beta(x)d_\Omega(x,\bar{y}) \le g(x,y)$ for all $y \in \Omega(x)$. By Lemma 3.2, $\bar{y} \in \Omega(x)$. Therefore $g(x,\bar{y}) \le g(x,y)$ for all $y \in \Omega(x)$ and $\bar{y} \in S(x)$. On the other hand, for any $\bar{y} \in S(x)$, Lemma 3.2 excludes the possibility of the existence of a $y' \notin \Omega(x)$ with

$$g(x,\bar{y}) + \pi(x,\bar{y}) = g(x,\bar{y}) > g(x,y) + \pi(x,y') \ge g(x,y') + \beta(x)d_\Omega(x,y);$$

therefore $\bar{y}$ actually minimizes $g(x,\cdot) + \pi(x,\cdot)$ over $y \in \mathbb{R}^m$. □

We should notice in the expression (3.2) that $\pi(x,y) = \infty$ if $\Omega(x) = \emptyset$ since the distance from any point to the empty set is infinity. The function π in Proposition 3.3 serves as an exact penalty function to the lower-level problem in the sense that the optimal solutions of the constrained problem consist of all the minimizers of a penalized objective function over $\mathbb{R}^m$. In the following corollaries, we will identify various conditions under which the appropriate exact penalty functions can be constructed.

COROLLARY 3.4. *Suppose* $\Omega(x)$ *is closed and* $g(x,\cdot)$ *satisfies a semi-Lipschitz condition relative to the set* $\Omega(x)$ *with modulus* $L(x)$ *for each* $x \in \operatorname{dom}\Omega$. *Let* $\pi : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}_+$ *and* $\beta : \mathbb{R}^n \to \mathbb{R}_+$ *be functions satisfying* $\beta(x) > L(x)$ *for all* x *and*

$$\operatorname{lev}_{\alpha\beta(x)}\pi := \{y \in \mathbb{R}^m \mid \pi(x,y) \le \alpha\beta(x)\} \subset \Omega(x) + \alpha\mathbb{B}_m \quad \forall \alpha \ge 0. \tag{3.3}$$

Then for any $x \in \operatorname{dom}\Omega$, *one has* $S(x) = \operatorname{argmin}\{g(x,y) + \pi(x,y) \mid y \in \mathbb{R}^m\}$.

Proof. It suffices to show that (3.2) and (3.3) are equivalent. In fact, from (3.2), for any $\alpha \ge 0$ and $y \in \operatorname{lev}_{\alpha\beta(x)}\pi$, one has

$$\beta(x)d_\Omega(x,y) \le \pi(x,y) \le \alpha\beta(x),$$

which implies $y \in \Omega(x) + \alpha\mathbb{B}$ and (3.3) follows. Conversely, assume (3.3), since

$$\Omega(x) + \alpha\mathbb{B} = \operatorname{lev}_\alpha \operatorname{dist}(\cdot \mid \Omega(x)) = \operatorname{lev}_{\alpha\beta(x)}(\beta(x)\operatorname{dist}(\cdot \mid \Omega(x))),$$

it follows that $\beta(x)\,\mathrm{dist}(\cdot \mid \Omega(x)) \le \pi(x,\cdot)$. □

In what follows, we shall say g is equi-Lipschitz continuous in y with constant L if $L > 0$ and for all $x \in \mathrm{dom}\,\Omega$, one has

$$|g(x,y) - g(x,y')| \le L|y - y'| \quad \forall\, y,\ y'.$$

COROLLARY 3.5. *If g is equi-Lipschitz continuous in y with constant L, then for any $\tau \ge L$, one has $S(x) = \mathrm{argmin}\{g(x,\cdot) + \tau d_\Omega(x,\cdot) \mid y \in \mathbb{R}^m\}$.*

Proof. This is a direct consequence of Proposition 3.3 by letting $L(x) \equiv L$ and $\beta(x) \equiv \tau$. □

For problems in which the graph of the constraint multifunction is a convex polyhedron:

$$\Omega(x) := \{y \mid Ax + By - c \le 0\}, \tag{3.4}$$

where $A \in \mathbb{R}^{k\times n}$, $B \in \mathbb{R}^{k\times m}$ and $c \in \mathbb{R}^k$, the following corollary justifies the conjecture that the penalty function can take a piecewise affine form. Define a piecewise affine function

$$\sigma_\Omega(x,y) := \max\{0, \langle a_1, x\rangle + \langle b_1, y\rangle - c_1, \ldots, \langle a_k, x\rangle + \langle b_k, y\rangle - c_k\}, \tag{3.5}$$

where a_i and b_i are row vectors of A and B, respectively, and c_i is the ith element of the vector c. One can see that $\sigma_\Omega(x,y) = 0$ if and only if $y \in \Omega(x)$, and that function λ defined by

$$\lambda(x) := \inf\{\sigma_\Omega(x,y) \mid d_\Omega(x,y) = 1\} \tag{3.6}$$

is bounded away from 0 by a constant meaning that there exists $\epsilon > 0$ with $\lambda(x) \ge \epsilon$ for all x.

COROLLARY 3.6. *Suppose $g(x,\cdot)$ satisfies a semi-Lipschitz condition relative to the set $\Omega(x)$ with modulus $L(x)$ for each $x \in \mathrm{dom}\,\Omega$ and Ω has the form of* (3.4). *Let $\beta : \mathbb{R}^n \to \mathbb{R}_+$ be a function satisfying $\beta(x) > L(x)$ for each $x \in \mathrm{dom}\,\Omega$. Then for any $x \in \mathrm{dom}\,\Omega$, one has $S(x) = \mathrm{argmin}\{g(x,y) + (\beta(x)/\lambda(x))\sigma_\Omega(x,y)) \mid y \in \mathbb{R}^m\}$.*

Proof. It suffices, by Proposition 3.3, to show that

$$\sigma_\Omega(x,y) \begin{cases} \ge \lambda(x) d_\Omega(x,y), & \text{if } y \notin \Omega(x), \\ = 0, & \text{otherwise.} \end{cases}$$

In fact, one has that $\sigma_\Omega(x,y) = 0$ if and only if $y \in \Omega(x)$ and due to the piecewise affine property of σ_Ω, for each x there exists y_1 with $d_\Omega(x,y_1) = 1$ such that

$$\sigma_\Omega(x,y) \ge \sigma_\Omega(x,y_1) d_\Omega(x,y) = \lambda(x) d_\Omega(x,y) \quad \forall\, y \notin \Omega(x).$$

We thus conclude the proof. □

As a general rule, the penalty function π does not always have to be a multiple of the distance function d_Ω as demonstrated in Corollary 3.6. The selection should be based on convenience as well as the properties that hold for the function $g + \pi$.

4. Optimality conditions to the cooperative problems. In this section, we develop optimality conditions in terms of a graph set of the solution multifunction to the lower-level problem. A linearly perturbed lower-level problem with a perturbation vector $v \in \mathbb{R}^m$ is considered first and then the original problem is treated as if it is

perturbed, but with $v = 0$. The advantage of this treatment is seen in the next section where the optimality conditions can be given in terms of the graphs of subdifferential multifunctions in both x and y variables, provided the lower-level problem is convex.

First of all, for a vector $v \in \mathbb{R}^m$, consider the linearly perturbed lower-level problem

$$\text{minimize} \quad g(x,y) - \langle v, y\rangle \quad \text{over} \quad y \in \mathbb{R}^m \quad \text{subject to} \quad y \in \Omega(x)$$

and denote the optimal solution set to this problem by $S_v(x)$. We thus have $S(x) = S_0(x)$ for each $x \in \mathbb{R}^n$. Define a multifunction (set-valued mapping) $F : \mathbb{R}^m \rightrightarrows \mathbb{R}^n \times \mathbb{R}^m$ by

$$F(v) := \operatorname{gph} S_v = \{(x,y) \mid y \in S_v(x)\}, \tag{4.1}$$

and we have $\operatorname{gph} S = F(0)$. Let $z := (x,y)$, the problem P_C can then be stated as

$$\text{minimize} \quad f(z) \quad \text{over} \quad z \in F(0) \cap D. \tag{P_0}$$

Let us recall for a moment the definitions of the *upper-Lipschitz* and *pseudo-Lipschitz continuities* of a multifunction.

DEFINITION 4.1. *A multifunction $\Gamma : \mathbb{R}^k \rightrightarrows \mathbb{R}^l$ is said to be upper-Lipschitz continuous at v if there exists a constant (modulus) $\kappa \geq 0$ and a neighborhood V of v such that*

$$\Gamma(v') \subset \Gamma(v) + \kappa|v - v'|\mathbb{B}_l \quad \forall\, v' \in V.$$

Γ is said to be pseudo-Lipschitz continuous at (v,z) for $z \in \Gamma(v)$ if there exist neighborhood V of v and W of z such that

$$\Gamma(v') \cap W \subset \Gamma(v'') + \kappa|v' - v''|\mathbb{B}_l \quad \forall\, v', \qquad v'' \in V.$$

In general, neither the pseudo-Lipschitz continuity of Γ at (v,z) for all $z \in \Gamma(v)$ implies the upper-Lipschitz continuity of Γ at v and vice versa. For example, the multifunction on $\mathbb{R}$

$$\Gamma(v) := \begin{cases} 1 + tv & \text{if } v > 0; \\ [-1,1] & \text{if } v = 0; \\ -1 + tv & \text{if } v < 0 \end{cases}$$

for some positive constant t is upper-Lipschitz at 0, but not pseudo-Lipschitz around $(0,u)$ for any $u \in \Gamma(0)$. Also, the multifunction $\Gamma(v) := \{0\} \cup \{1/v \mid v \neq 0\}$ defined on $\mathbb{R}$ is pseudo-Lipschitz around $(0,u)$ for all $u \in \Gamma(0)$, but not upper-Lipschitz at 0. (For more details on pseudo-Lipschitz multifunctions see, for example, [16] and [22].) One important class of upper-Lipschitz multifunctions is that of polyhedral mappings in the sense of Robinson [18], [19]: the graph of a polyhedral multifunction consists of a union of finitely many polyhedral convex sets. For a linear-quadratic lower-level problem, the solution multifunction S_v becomes polyhedral and therefore the multifunction F in (4.1) is upper-Lipschitz.

For a two-level hierarchical optimization problem, if F is upper-Lipschitz and the upper-level objective function f satisfies certain growth conditions, then the effect of adding a perturbation term to the lower-level objective is equivalent to that of adding a norm function of the perturbation vector to the upper-level objective. In an economic and maybe somewhat less rigorous sense, the outcome remains the same if the follower's solution is altered by a small amount, while the leader bears an additional cost. The next proposition justifies this.

For a positive constant r, let (P_r) be the following problem with the perturbed constraints:

$$(P_r) \qquad \text{minimize } f(z)+r|v| \text{ over } (z,v) \text{ subject to } z \in F(v)\cap D,\ v\in V$$

for some neighborhood V of $0 \in \mathbb{R}^m$. We establish an equivalence between problems (P_0) and (P_r) under certain conditions.

PROPOSITION 4.2. *Suppose the multifunction $F\cap D : v \mapsto F(v)\cap D$ is upper-Lipschitz continuous at $v=0$ with constant κ and the upper-objective function f satisfies a semi-Lipschitz condition relative to $F(0)\cap D$ with modulus M. Then for any $r>\kappa M$, problems (P_0) and (P_r) are equivalent in the following sense*: (a) *if $\bar z$ solves (P_0) then $(\bar z,0)$ solves (P_r), and* (b) *if $(\bar z,\bar v)$ solves (P_r) then $\bar v=0$ and $\bar z$ solves (P_0).*

Proof. Let $\bar z$ be a solution to (P_0), one has

$$f(\bar z)\le f(z') \quad \forall\, z'\in F(0)\cap D.$$

For any $v\in V$ and $z\in F(v)\cap D$, if $z\in F(0)$, then

$$f(\bar z)\le f(z)\le f(z)+r|v|.$$

If $z\notin F(0)$, by the upper-Lipschitz continuity of $F\cap D$, there exists $z_0\in F(0)\cap D$ such that $|z-z_0|\le\kappa|v|$ and, by the semi-Lipschitz condition of f, one has

$$f(\bar z)\le f(z_0)\le f(z)+M|z_0-z|\le f(z)+\kappa M|v|<f(z)+r|v|.$$

Therefore $(\bar z,0)$ solves (P_r) for some V.

Conversely, if $(\bar z,\bar v)$ solves (P_r) for $r>\kappa M$, one has $\bar z\in F(\bar v)\cap D$ and $f(\bar z)+r|\bar v|\le f(z)+r|v|$ for all $z\in F(v)\cap D$. In particular,

$$f(\bar z)+r|\bar v|\le f(z) \quad \forall\, z\in f(0)\cap D. \tag{4.2}$$

If $\bar z\in F(0)$, one obtains $\bar v=0$ by letting $z=\bar z$ in (4.2). Otherwise if $\bar z\notin F(0)$, by the semi-Lipschitz condition of f and the upper-Lipschitz continuity of $F\cap D$, there exists $z_0\in F(0)\cap D$ such that

$$r|\bar v|\le f(z_0)-f(\bar z)\le M|z_0-\bar z|\le\kappa M|\bar v|.$$

This yields $\bar v=0$ since $r>\kappa M$ and $\bar z$ solves (P_0). □

To state our optimality conditions that appear later, we need to review some of the definitions and the properties of the *normal cones* to a subset and the *subgradients* of a lower-semicontinuous function in the sense of Mordukhovich [15]. One of the reasons for using these normal cones instead of Clarke normal cones is that we are primarily interested in the normals to the graph of the multifunction F. In general, the set $\operatorname{gph} F$ is not Clarke regular [5] (also referred to as being tangentially regular) and the Clarke normal cones to $\operatorname{gph} F$ at most points turn out to be "too large to be sensitive enough." Consider a simple example where $g(x,y)=x+|y|$ and $\Omega(x)=\mathbb{R}$ for $x,y\in\mathbb{R}$. We have

$$F(v)=\begin{cases}[0,\infty) & \text{for } v=1,\\ \{0\} & \text{for } v\in[-1,1],\\ (-\infty,0] & \text{for } v=-1.\end{cases}$$

The Clarke's normal cone to $\operatorname{gph} F$ at every corner point $(x, y) \in \operatorname{gph} F$ is the whole plane and a line elsewhere.

Let C be a nonempty, closed subset in a finite dimensional space and $\bar{z} \in C$. The *normal cone* to C at $\bar{z}$ is the closed cone

$$N(\bar{z}|C) := \limsup_{z \xrightarrow{C} \bar{z}} T^{\circ}(z|C),$$

where $T^{\circ}(z|C)$ is the polar to Bouligard *contingent cone* to C at $\bar{z}$ and "limsup" is the *upper limit* of multifunctions in the sense of Kuratowski–Painlevé. Denote the Clarke normal cone to C at $\bar{z}$ by $\overline{N}(\bar{z}|C)$; it is well known that $N(\bar{z}|C)$ may not be convex and $\overline{N}(\bar{z}|C) = \operatorname{cl}\operatorname{co} N(\bar{z}|C)$ where "cl co" denotes the *closed convex hull.* If C happens to be Clarke regular [5], then $\overline{N}(\bar{z}|C)$ and $N(\bar{z}|C)$ coincide. One can imagine that, in general, $N(\bar{z}|C)$ is more sophisticated in structure than $\overline{N}(\bar{z}|C)$. Let h be a lower semicontinuous function on $\mathbb{R}^k$, $\bar{z} \in \mathbb{R}^k$ and $\operatorname{epi} h := \{(z, \alpha) \mid h(z) \leq \alpha\}$ be the epigraph of h, the set of (Mordukhovich) *subgradients* of h at $\bar{z}$ is the set

$$\partial h(\bar{z}) := \{z^* \mid (z^*, -1) \in N((\bar{z}, h(\bar{z}))|\operatorname{epi} h)\} \tag{4.3}$$

and the *singular subgradient* of h at $\bar{z}$ is the set

$$\partial^{\infty} h(\bar{z}) := \{z^* \mid (z^*, 0) \in N((\bar{z}, h(\bar{z}))|\operatorname{epi} h)\}.$$

The subgradient (4.3) coincides with the Clarke generalized gradient if h is Clarke regular and with the subgradient in convex analysis if h is convex. (For more details on the normal cones and subgradient of Mordukhovich, see [15]–[17].) As far as the subgradients are concerned, Rockafellar, Clarke, and Loewen have also dealt with the same objects from a different aspect by considering the set of all *limiting proximal subgradients* of h at $\bar{z}$ (see for example [6], [21]).

The subgradients (4.3) have many important properties. For example, if C is a closed subset in $\mathbb{R}^k$ and $\bar{z} \in C$, one has $\partial \delta_C(\bar{z}) = N(\bar{z}|C)$; if $\bar{z}$ minimizes a lower-semicontinuous function h over $C \subset$ and the only $v \in \partial^{\infty} h(\bar{z})$ with $-v \in N(\bar{z}|C)$ is $v = 0$, it is necessary that $0 \in \partial h(\bar{z}) + N(\bar{z}|C)$; $\partial^{\infty} h(\bar{z}) = \{0\}$ if and only if h is locally Lipschitz at $\bar{z}$. From these facts and Proposition 4.2, we can derive a necessary optimality condition for the solutions to problem (P_C) as follows.

PROPOSITION 4.3. *If $\bar{z} = (\bar{x}, \bar{y})$ is a solution to (P_C) and, in addition to the assumptions of Proposition 4.2, assume that the only z_i^* for $i = 1, 2, 3$ such that*

$$z_1^* \in \partial^{\infty} f(\bar{z}), \quad z_2^* \in N(\bar{z}|D), \quad (0, z_3^*) \in N((0, \bar{z})|\operatorname{gph} F) \text{ with } z_1^* + z_2^* + z_3^* = 0 \tag{4.4}$$

are $z_1^* = z_2^* = z_3^* = 0$. Then for any $r > \kappa M$ with κ being the upper-Lipschitz constant of F and M the semi-Lipschitz modulus of f, there exist vectors $\eta \in r\mathbb{B}_m$ and $q \in \partial f(\bar{z}) + N(\bar{z}|D)$ such that

$$-(\eta, q) \in N((0, \bar{z})|\operatorname{gph} F). \tag{4.5}$$

Proof. From Proposition 4.2, $(0, \bar{x}, \bar{y})$ minimizes $f(x, y) + r|v|$ over $(x, y) \in F(v) \cap D$ and $v \in V$. Since $\partial_v\{r|v|\} = r\mathbb{B}_m$ at $v = 0$ and (4.4) guarantees the conditions in Theorem 3.1 of [16], it follows that

$$(0, 0) \in r\mathbb{B}_m \times \partial f(\bar{z}) + \{0\} \times N(\bar{z}|D) + N((0, \bar{z})|\operatorname{gph} F).$$

We conclude the proof. □

It should be pointed out that the condition (4.4) is satisfied if in particular f is Lipschitz continuous and $D = \mathbb{R}^n \times \mathbb{R}^m$. In the next section, where we assume the lower-level problem $(Q(x))$ to be convex, the multifunction F is identified with the partial subdifferential of g with respect to y. In connection with the subdifferential of g with respect to (x, y), we have the following proposition under the general setting. A multifunction $\Gamma\colon \mathbb{R}^k \rightrightarrows \mathbb{R}^l$ is said to be locally bounded at v if there exists a neighborhood N of v such that $\Gamma(N)$ is bounded.

PROPOSITION 4.4. *If $\bar{z} = (\bar{x}, \bar{y})$ is a solution to (P_C) and, in addition to the assumptions of Proposition* 4.2, *the following conditions are satisfied, then what follows is true.*

(a) *There exists a multifunction $G : \mathbb{R}^n \times \mathbb{R}^m \rightrightarrows \mathbb{R}^n \times \mathbb{R}^m$ such that* $\operatorname{gph} F$ *is the projection of* $\operatorname{gph} G$ *under $(u, v, x, y) \mapsto (v, x, y)$.*

(b) *The only z_i^* for $i = 1, 2, 3$ satisfying for some $w \in \mathbb{R}^n \times \{0\}$ with $\bar{z} \in G(w)$ and*

$$\begin{aligned} &z_1^* \in \partial^\infty f(\bar{z}),\ z_2^* \in N(\bar{z}|D), \\ &(0, z_3^*) \in N((w, \bar{z})|\operatorname{gph} G) \text{ with } z_1^* + z_2^* + z_3^* = 0 \end{aligned} \tag{4.6}$$

are $z_1^* = z_2^* = z_3^* = 0$;

(c) *The multifunction $(v, x, y) \mapsto \{u \mid (u, v) \in G(x, y)\}$ is locally bounded at $(0, \bar{z})$.*

Then for any $r > \kappa M$, there exist vectors $\eta \in r\mathbb{B}_m$, $\bar{u} \in \mathbb{R}^n$ with $(\bar{u}, 0) \in G(\bar{z})$ and $q \in \partial f(\bar{z}) + N(\bar{z}|D)$ such that

$$-(p, q) \in N((\bar{w}, \bar{z})|\operatorname{gph} G) \tag{4.7}$$

where $p := (0, \eta)$ and $\bar{w} := (\bar{u}, 0)$.

Proof. From Corollary 3.1.2 of [21] (or Theorem 4.1 of [16]), we have for any $(v, x, y) \in \operatorname{gph} F$ that

$$N((v, x, y)|\operatorname{gph} F) \subset \bigcup_{u:(x,y)\in G(u,v)} \{(v', x', y') \mid (0, v', x', y') \in N((u, v, x, y)|\operatorname{gph} G)\} \tag{4.8}$$

provided (c) is satisfied. It follows that $(0, z_3^*) \in N((0, \bar{z})|\operatorname{gph} F)$ implies the existence of $w \in \mathbb{R}^n \times \{0\}$ such that $\bar{z} \in G(w)$ and $(0, z_3^*) \in N((w, \bar{z})|\operatorname{gph} G)$. Therefore the conditions in Proposition 4.3 are satisfied. Therefore, combining (4.5) and (4.8) gives (4.7). □

5. Optimality conditions when the lower-level problem is convex. Here we specialize the optimality conditions obtained in the previous section to the problems with convex lower-level problems in the sense that for each x the lower-level objective function $g(x, \cdot)$ and the constraint set $\Omega(x)$ are convex. First, let us give a definition on the *strong minimum* of a extended-real-valued function $\theta : \mathbb{R}^m \to \overline{\mathbb{R}}$.

DEFINITION 5.1. *The function θ is said to have strong minimum with modulus κ if the inverse of the subdifferential multifunction of θ is upper-Lipschitz continuous at* 0 *with modulus κ.*

EXAMPLE. *If $C \subset \mathbb{R}^k$ is nonempty and convex, then the distance function* $\operatorname{dist}(\cdot|C)$ *and* $(\frac{1}{2})\operatorname{dist}^2(\cdot|C)$ *have strong minimum with modulus* 0 *and* 1, *respectively.*

Proof. (i) Let Γ_C be the inverse mapping of $\partial \operatorname{dist}(\cdot|C)$, it is obvious that $\operatorname{cl} C \subset \Gamma_C(0)$. On the other hand, let $\varepsilon < 1$, we can show that $\Gamma_C(v) \subset \operatorname{cl} C$ for all $v \in \varepsilon\mathbb{B}$. In fact, if there were $y' \in \Gamma_C(v)$ but $y' \notin \operatorname{cl} C$, one would have by (2.5.4) of [5] that

$$v = \nabla \operatorname{dist}(y'|C) = \frac{y' - c_0}{|y' - c_0|},$$

where c_0 is the projection of y' on $\operatorname{cl} C$. This contradicts with the fact that $v \in \epsilon\mathbb{B}$. It follows that

$$\Gamma_C(v) \subset \operatorname{cl} C \subset \Gamma_C(0) \quad \forall v \in \varepsilon\mathbb{B},$$

so Γ_C is upper-Lipschitz continuous with modulus 0 at 0.

(ii) Let $\widetilde{\Gamma}_C$ be the inverse mapping of $\partial(\frac{1}{2})\operatorname{dist}^2(\cdot|C)$, $v \in \mathbb{R}^k$ and $\bar{y} \in \widetilde{\Gamma}_C(v)$. If $\bar{y} \in \operatorname{cl} C$ then $0 \in \partial(\frac{1}{2})\operatorname{dist}^2(\bar{y}|C)$ so $\bar{y} \in \widetilde{\Gamma}_C(0)$. Otherwise, if $\bar{y} \notin \operatorname{cl} C$, one has $\operatorname{dist}(\bar{y}|C) = |v|$ and therefore $\bar{y} \in \widetilde{\Gamma}_C(0) + |v|\mathbb{B}$. We conclude that $\widetilde{\Gamma}_C$ is upper-Lipschitz continuous with modulus 1 at 0. □

In an intuitive way, what the strong minimum condition of θ demands is that θ must grow with a positive curvature (infinity if θ is nondifferentiable) when the argument moves across the boundary of the set of all minimizers. A piecewise linear-quadratic function has necessarily strong minimum since the subdifferential multifunction of such a function is polyhedral and so is its inverse mapping. A function with *sharp minimum* on its argmin set in the sense of Ferris [7] is convex and has strong minimum. Also, strongly convex functions are in particular convex and have strong minimum. In fact, we can show that a convex function h with $\operatorname{argmin}(h) \neq \emptyset$ has strong minimum if and only if there exists $\varepsilon > 0$ and $c \geq 0$ such that

$$h(y) \geq \inf h + c\operatorname{dist}^2(y|\operatorname{argmin}(h)) \quad \forall y \in \operatorname{argmin}(h) + \varepsilon\mathbb{B}.$$

The proof can be found in [29].

The following assumptions will be needed soon.

Assumption A. (i) The lower-level objective function g satisfies a semi-Lipschitz condition in y relative to $\Omega(x)$ with modulus $L(x)$ and $\Omega(x)$ is closed for each $x \in \operatorname{dom}\Omega$.

(ii) The upper-level objective function f satisfies a semi-Lipschitz condition relative to $F(0) \cap D$ with modulus M.

LEMMA 5.2. *Assume* A(i). *If π is a function satisfying* (3.2) *and $(g + \pi)(x, \cdot)$ is convex having strong minimum with modulus κ for all $x \in \operatorname{dom}\Omega$, then the multifunction F defined in* (4.1) *is upper-Lipschitz continuous in (x, y) at $(0, 0)$ with constant κ.*

Proof. Let $\Gamma_x(\cdot)$ be the inverse of the subdifferential multifunction of $\partial_y(g(x, \cdot) + \pi(x, \cdot))$. Then there exists a neighborhood V of $0 \in \mathbb{R}^m$ such that all $v \in V$, one has

$$\begin{aligned}
F(v) &= \{(x, y) \mid y \in S_v(x)\} \\
&= \{(x, y) \mid y \in \operatorname{argmin}\{g(x, \cdot) + \pi(x, \cdot) - \langle v, \cdot\rangle\}\} \quad \text{(Proposition 3.3.)} \\
&= \{(x, y) \mid y \in \Gamma_x(v)\} \quad \text{(convexity of } g + \pi \text{ in } y) \\
&= \{(x, y) \mid y \in \Gamma_x(0) + \kappa|v|\mathbb{B}_m\} \quad \text{(strong minimum condition of } g + \pi \text{ in } y) \\
&= F(0) + \{0\} \times \kappa|v|\mathbb{B}_m \subset F(0) + \kappa|v|\mathbb{B}_{n+m}.
\end{aligned}$$

Therefore F is upper-Lipschitz continuous at 0 with the constant κ. □

PROPOSITION 5.3. *If $\bar{z} := (\bar{x}, \bar{y})$ is a solution to (P_C) and, in addition to Assumption* A, *assume the following are true*:

(a) *π is a function such that* (3.2) *is satisfied and $g + \pi$ is convex having strong minimum in y with modulus κ*;

(b) *the only z_i^* for $i = 1, 2, 3$ such that, for some $w \in \mathbb{R}^n \times \{0\}$,*

$$z_1^* \in \partial^\infty f(\bar{z}),\ z_2^* \in N(\bar{z}|D),\ (0, z_3^*) \in N((0, \bar{z})|\operatorname{gph} F) \text{ with } z_1^* + z_2^* + z_3^* = 0$$

are $z_1^* = z_2^* = z_3^* = 0$.

Then for any $r > \kappa M$, there exist vectors $\eta \in r\mathbb{B}_m$ and $q \in \partial f(\bar{z}) + N(\bar{z}|D)$ such that

$$-(q, \eta) \in N((\bar{z}, 0)|\operatorname{gph} \partial_y(g + \pi)). \tag{5.1}$$

Proof. From the convexity of $(g + \pi)(x, \cdot)$ and Proposition 3.3, one has that y solves $Q(x)$ if and only if $0 \in \partial_y(g + \pi)(x, y)$ and therefore $F(0) = \{(x, y) \mid 0 \in \partial_y(g + \pi)(x, y)\}$. From Lemma 5.2, F is upper-Lipschitz at 0 and the conditions in Proposition 4.3 are satisfied. The conclusion then follows from Proposition 4.3. □

Recall that if $(g + \pi)$ is locally Lipschitz and Clarke regular at $\bar{z} = (\bar{x}, \bar{y})$, then the set $\partial_y(g + \pi)(\bar{x}, \bar{y})$ is compact and is given as the image of $\partial(g + \pi)(\bar{x}, \bar{y})$ under the projection $(u, v) \mapsto v$ [5, (2.3.15), (2.3.16)]. If we define a multifunction $G : \mathbb{R}^n \times \mathbb{R}^m \rightrightarrows \mathbb{R}^n \times \mathbb{R}^m$ by

$$G(u, v) := \{(x, y) \mid (u, v) \in \partial(g + r\pi)(x, y)\}, \tag{5.2}$$

by applying Proposition 4.4, we are able to obtain an optimality condition in terms of the subgradients of $g + \pi$ instead of the partial subgradients.

PROPOSITION 5.4. *If $\bar{z} := (\bar{x}, \bar{y})$ is a solution to (P_C) and, in addition to Assumption* A, *assume the following are true*:

(a) *π is a function such that* (3.2) *is satisfied and $g + \pi$ is convex having strong minimum in y with modulus κ*;

(b) *$g + \pi$ is locally Lipschitz and Clarke regular in (x, y)*;

(c) *the only z_i^* for $i = 1, 2, 3$ such that, for some $w \in \mathbb{R}^n \times \{0\}$,*

$$z_1^* \in \partial^\infty f(\bar{z}),\ z_2^* \in N(\bar{z}|D),\ (z_3^*, 0) \in N((\bar{z}, w)|\operatorname{gph} \partial(g + \pi)) \text{ with } z_1^* + z_2^* + z_3^* = 0$$

are $z_1^* = z_2^* = z_3^* = 0$.

Then for any $r > \kappa M$, there exist vectors $\eta \in r\mathbb{B}_m$, $\bar{u} \in \mathbb{R}^n$ and $q \in \partial f(\bar{z}) + N(\bar{z}|D)$ such that

$$-(q, p) \in N((\bar{z}, \bar{w})|\operatorname{gph} \partial(g + \pi)),$$

where $p := (0, \eta)$ and $\bar{w} := (\bar{u}, 0)$.

Proof. With G as defined in (5.2), the locally Lipschitz continuity of $\partial(g + \pi)$ guarantees the locally boundedness of (4.6) while the Clarke regularity implies the projection stated prior to the proposition. The rest of the assumptions in Proposition 4.4 are easily verified. Combining (4.7) with (5.2), we conclude the proof. □

The following corollaries deal with various special cases of Propositions 5.3 and 5.4.

COROLLARY 5.5. *If $\bar{z} := (\bar{x}, \bar{y})$ is a solution to (P_C) and the following assumptions are satisfied*:

(a) *g and f are Lipschitz continuous in (x,y) with modulus L and M, respectively*;

(b) *π is a locally Lipschitz function satisfying* (3.2), *$g+\pi$ is convex having strong minimum in y with modulus κ and is Clarke regular in (x,y)*;

(c) *the only z^* satisfying, for some $w \in \mathbb{R}^n \times \{0\}$ such that $w \in \partial(g+\pi)$, $z^* \in N(\bar{z}|D)$ and $(z^*,0) \in N((\bar{z},w)|\operatorname{gph}\partial(g+\pi))$ is $z^*=0$.*

Then for any $r > \kappa M$, there exist vectors $\eta \in r\mathbb{B}_m$, $\bar{u} \in \mathbb{R}^n$ and $q \in \partial f(\bar{z}) + N(\bar{z}|D)$ such that

$$-(q,p) \in N((\bar{z},\bar{w})|\operatorname{gph}\partial(g+\pi)),$$

where $p := (0,\eta)$ and $\bar{w} := (\bar{u},0)$.

Proof. This is a special case of Proposition 5.4 with $\partial^\infty(\bar{z}) = \{0\}$ and the Lipschitz continuities of f and g replacing the semi-Lipschitz conditions. □

COROLLARY 5.6. *If $\bar{z} = (\bar{x},\bar{y})$ is a solution to (P_C) with $D = \mathbb{R}^n \times \mathbb{R}^m$ and the following assumptions are satisfied*:

(a) *g is equi-Lipschitz in y with modulus L and f is Lipschitz continuous with modulus M*;

(b) *$g + \tau d_\Omega$ is convex having strong minimum in y with modulus κ for some $\tau > L$ and is Clarke regular in (x,y)*;

(c) *Ω is pseudo-Lipschitz continuous at each (x,y) for all $y \in \Omega(x)$*;

then for any $r > \kappa M$, there exist vectors $\eta \in r\mathbb{B}_m$, $\bar{u} \in \mathbb{R}^n$ and $q \in \partial f(\bar{z})$ such that

$$-(q,p) \in N((\bar{z},\bar{w})|\operatorname{gph}\partial(g+\tau d_\Omega)),$$

where $p := (0,\eta)$ and $\bar{w} := (\bar{u},0)$.

Proof. The pseudo-Lipschitz continuity of Ω implies the locally Lipschitz continuity of d_Ω at all (x,y) (Theorem 2.3 of [21]). The claim then follows immediately from Corollary 3.5 and Proposition 5.4 by letting $\pi = \tau d_\Omega$ and $L(x) \equiv L$. □

COROLLARY 5.7. *If $\bar{z} = (\bar{x},\bar{y})$ is a solution to (P_C) with $D = \mathbb{R}^n \times \mathbb{R}^m$, Ω has the polyhedral form* (3.4), *and the following assumptions are satisfied*:

(a) *g is equi-Lipschitz in y with modulus L and f is Lipschitz continuous with modulus M*;

(b) *$g + \tau\sigma_\Omega$ is convex having strong minimum in y with modulus κ for some $\tau > L$ and is Clarke regular in (x,y)*;

then for any $r > \kappa M$, there exist vectors $\eta \in r\mathbb{B}_m$, $\bar{u} \in \mathbb{R}^n$ and $q \in \partial f(\bar{z})$ such that

$$-(q,p) \in N((\bar{z},\bar{w})|\operatorname{gph}\partial(g+\tau\sigma_\Omega)), \tag{5.3}$$

where $p := (0,\eta)$, $\bar{w} := (\bar{u},0)$, and σ_Ω is given in (3.5).

Proof. The multifunction Ω defined by (3.4) has convex graph therefore is pseudo-Lipschitz. With $\partial^\infty f(\bar{z}) = \{0\}$ and $N(\bar{z}|D) = \{0\}$, the condition (c) in Proposition 5.4 is naturally satisfied. Notice that $\lambda(x)$ is bounded away from 0 by a constant for all x and thus $L/\lambda(x)$ is bounded above. Therefore τ does exists. The remainder are the direct consequences of Corollary 3.6 and Proposition 5.4. □

One of the advantages of the optimality conditions in Propositions 5.3 and 5.4 is that they can be expressed in terms of the *second-order subdifferentials* described in [15]–[17] and thus the related calculus developed can be used. Let $H: \mathbb{R}^k \rightrightarrows \mathbb{R}^k$ be a multifunction and $w \in H(z)$. The coderivative of H at (z,w) is the multifunction $p \mapsto D^*H(z,w)(p)$ defined by

$$D^*H(z,w)(p) := \{q \mid (q,-p) \in N((z,w)|\operatorname{gph}H)\}.$$

For a function $\varphi : \mathbb{R}^k \to \overline{\mathbb{R}}$, $z \in \operatorname{dom}\varphi$ and $w \in \partial\varphi(z)$ with $\partial\varphi$ as defined in (4.3). The multifunction $\partial^2\varphi(z,w) : \mathbb{R}^k \rightrightarrows \mathbb{R}^k$ defined by

$$\partial^2\varphi(z,w)(p) := D^*\partial\varphi(z,w)(p) = \{q \mid (q,-p) \in N((z,w)|\operatorname{gph}\partial\varphi)\} \tag{5.4}$$

is called the second-order subdifferential of φ at z relative to w. If φ happens to be continuously twice differentiable, then $\partial^2\varphi(z)(p) = \nabla^2\varphi(z)p$ for all p where $\nabla^2\varphi$ is the usual Hessian matrix.

The following corollary can be derived easily from Proposition 5.4 but uses the notions of the second-order subdifferentials.

PROPOSITION 5.8. *If $\bar{z} = (\bar{x},\bar{y})$ is a solution to (P_C) and the following assumptions are satisfied*:

(a) *for each x, $g(x,\cdot)$ satisfies a semi-Lipschitz condition relative to $\Omega(x)$ with modulus $L(x)$ and f is Lipschitz continuous in (x,y) with modulus M*;

(b) *π is a function such that (3.2) is satisfied and $g+\pi$ is convex having strong minimum in y with modulus κ*;

(c) *$g+\pi$ is locally Lipschitz and Clarke regular in (x,y)*;

(d) *for any $w \in \mathbb{R}^n \times \{0\}$ with $w \in \partial(g+\pi)$, the qualification condition*

$$\partial^2(g+\pi)(w,\bar{z}) \cap [-N(\bar{z}|D)] = \{0\} \tag{5.5}$$

is satisfied;

then for any $r > \kappa M$, there exist vectors $p \in \{0\} \times r\mathbb{B}_m$ and $\bar{w} \in \mathbb{R}^n \times \{0\}$ such that

$$0 \in \partial f(\bar{z}) + N(\bar{z}|D) + \partial^2(g+\pi)(\bar{z},\bar{w})(p). \tag{5.6}$$

Proof. It suffices to show that the condition (d) implies (c) of Proposition 5.4. In fact, since $\partial^\infty f(\bar{z}) = \{0\}$, $z_1^* = 0$. Let $z^* := z_2^* = -z_3^*$, one has

$$z^* \in N(\bar{z}|D) \quad \text{and} \quad (-z^*,0) \in N((\bar{z},w)|\operatorname{gph}\partial(g+\pi)).$$

Therefore $z^* = 0$ due to (5.4) and (d). □

As an example for applications of Corollary 5.6, consider a two-level problem with the piecewise linear-quadratic programming problem at the lower level where g is convex and piecewise linear-quadratic in the sense of Rockafellar [20] and Ω has the form of (3.4) with bounded graph. Then $g+\tau\sigma_\Omega$ is convex having strong minimum in y for all x where τ is a constant greater than the Lipschitz constant of $g(x,\cdot)$. Therefore the conditions of Corollary 5.6 are satisfied if f is Lipschitz. Due to the structure of σ_Ω, the set $\operatorname{gph}\partial(g+\tau\sigma_\Omega)$ in (5.3) can be calculated by using the formulas in [5, (2.5.1) and (2.3.12)]: let $d_i := (a_i,b_i)$, g_j be the jth piece of g, $J(\bar{z})$ the set of indices of the active pieces of g at $\bar{z}$: $J(\bar{z}) := \{j \mid g(\bar{z}) = g_j(\bar{z})\}$ and $I(\bar{z})$ the set of indices of active constraints. Then

$$\begin{aligned}
&\operatorname{gph}\partial(g+\tau\sigma_\Omega)(\bar{z}) \\
&= \operatorname{co}\{\lim_{z_j\to\bar{z}} \nabla g_j(z_j) + d_i \mid j \in J(\bar{z}), i \in I(\bar{z})\} \\
&= \left\{ \sum_{j\in J(\bar{z})} \mu_j \lim_{z_j\to\bar{z}} \nabla g_j(z_j) + \sum_{i\in I(\bar{z})} d_i \,\middle|\, \sum_{j\in J(\bar{z})} \mu_j + \sum_{i\in I(\bar{z})} 0 \le \lambda_i \le 1, \mu_j \ge 0 \right\} \\
&= \left\{ \sum_{j\in J(\bar{z})} \lim_{z_j\to\bar{z}} \mu_j \nabla g_j(z_j) + \begin{bmatrix} A^* \\ B^* \end{bmatrix} \lambda \mid (\mu,\lambda) \in T(\bar{z}) \right\},
\end{aligned}$$

where $T(\bar{z})$ is the set defined by

$$T(\bar{z}) := \left\{ (\mu, \lambda) \mid \lambda^\top \left(\begin{bmatrix} A \\ B \end{bmatrix} \bar{z} - c \right) = 0, \sum_{j \in J(\bar{z})} \mu_j + \sum_{i \in I(\bar{z})} \lambda_i \leq 1, \mu \geq 0, \lambda \geq 0 \right\}$$

and by $\lambda > 0$ we mean all the elements of λ are positive. The limit $\lim_{z_j \to \bar{z}} \nabla g_j(z_j)$ should be understood as taking place when z_j approaches $\bar{z}$ from the cell corresponding to the piece g_j. From this and (5.3), a scheme may be constructed to calculate a candidate of a solution to a given two-level problem.

For an even simpler example where, in addition to the assumptions of Proposition 5.8, g happens to be continuously twice differentiable, f is continuously differentiable and $\Omega(x) = \mathbb{R}^m$, it is easy to see that (5.6) yields the conditions one would get by using Lagrange multiplier rule due to the fact that $\partial^2 g(\bar{z}, \nabla g(\bar{z}))(p) = \nabla^2 g(\bar{z})p$.

REFERENCES

[1] J. P. AUBIN, *Contingent derivatives of set-valued maps and existence of solutions to nonlinear inclusions and differential inclusions*, Mathematical Analysis and Applications Part A, Adv. in Math. Suppl. Stud., 7a, Academic Press, New York, London, 1981, pp. 159–229.

[2] ———, *Lipschitz behavior of solutions to convex minimization problems*, Math. Oper. Res., 9 (1984), pp. 87–111.

[3] J. P. AUBIN AND I. EKELAND, *Applied Nonlinear Analysis*, Wiley Intersciences, New York, 1984.

[4] J. F. BARD, *Convex two-level optimization*, Math. Programming, 40 (1988), pp. 15–27.

[5] F. H. CLARKE, *Optimization and Nonsmooth Analysis*, Wiley Intersciences, New York, 1983.

[6] ———, *Applications of proximal subgradients*, Nonsmooth Optimization and Related Topics, F. H. Clark, ed., Plenum, New York, London, 1989.

[7] M. C. FERRIS, *Finite termination of the proximal point algorithm*, Math. Programming, 50 (1991), pp. 359–366.

[8] J. FORTUNY-AMAT AND B. MCCARL, *A representation and economic interpretation of a two-level programming problem*, J. Oper. Res. Soc., 32 (1981), pp. 783–792.

[9] W. W. HOGAN, *Point-to-set maps in mathematical programming*, SIAM Rev., 15 (1972), pp. 591–603.

[10] A. D. IOFFE, *Approximate subdifferentials and applications* 3: *the metric theory*, Mathematika, 36 (1989), pp. 1–38.

[11] Y. ISHIZUKA AND E. AIYOSHI, *Double penalty method for bilevel optimization problems*, Ann. Oper. Res., 34 (1992), pp. 73–88.

[12] D. KLATTE, *Lipschitz continuity of infima and optimal solutions in parametric optimization: the polyhedral case*, Parametric Optimization and Related Topics, Math. Res., 35 (1987), pp. 229–248.

[13] P. LORIDAN AND J. MORGAN, *Approximate solutions for two-level optimization problems*, Numer. Math., 84 (1988), pp. 181–196.

[14] B. LUCCHETTI, F. MIGNANEGO, AND G. PIERI, *Existence theorems of equilibrium points in Stackelberg games with constraints*, Optimization, 18 (1987), pp. 857–866.

[15] B. S. MORDUKHOVICH, *Optimization of differential inclusions via finite differences*, Proc. 30th IEEE Conf. Decision and Control, Brighton, England, 1991, pp. 257–262.

[16] ———, *Sensitivity analysis in nonsmooth optimization*, in Theoretical Aspects of Industrial Design, D. A. Field, and V. Komkov, eds., Society for Industrial and Applied Mathematics, Philadelphia, PA, 1992, pp. 32–46.

[17] ———, *On variational analysis of differential inclusions*, Optimization and Nonlinear Analysis, A. Ioffe, L. Marcus, and S. Reich, eds., Longman Scientififc and Technical, U.K., 1992, pp. 199–214.

[18] S. M. ROBINSON, *Some continuity properties of polyhedral multifunctions*, Math. Programming Study, 14 (1981), pp. 206–214.

[19] ———, *Generalized equations and their solutions, Part* I: *applications to nonlinear programming*, Math. Programming Study, 10 (1979), pp. 128–141.

[20] R. T. ROCKAFELLAR, *Linear-quadratic programming and optimal control*, SIAM J. Control Optim., 25 (1987), pp. 781–814.

[21] ———, *Extensions of subgradient calculus with applications to optimization*, Nonlinear Anal., 9 (1985), pp. 167–184.

[22] ———, *Lipschitz properties of multifunctions*, Nonlinear Anal., 9 (1985), pp. 867–885.

[23] - ——, *Nonsmooth analysis and parametric optimization*, Methods of Nonconvex Analysis, A. Cellina, ed., Springer-Verlag Lecture Notes in Math., No. 1446 (1990), pp. 137–151.

[24] H. VON STACKELBERG, *The Theory of The Market Economy*, Oxford University Press, Oxford, 1952.

[25] J. S. TREIMAN, *Characterization of Clarke's tangent and normal cones in finite and infinite dimensions*, Nonlinear Anal., 7 (1983), pp. 771–783.

[26] D. E. WARD AND J. M. BORWEIN, *Nonsmooth calculus in finite dimensions*, SIAM J. Control Optim., 26 (1987), pp. 1312–1340.

[27] D. E. WARD, *Exact penalties and sufficient conditions for optimality in nonsmooth optimization*, J. Optim. Theory Appl., 57 (1988), pp. 485–499.

[28] U. P. WEN AND S. T. SHU, *Linear bi-level programming problems—a review*, J. Optim. Res. Soc., 42 (1991), pp. 125–133.

[29] R. ZHANG AND J. TREIMAN, *Upper-Lipschitz multifunctions and inverse subdifferentials*, Nonlinear Anal., to appear.

SIAM J. OPTIMIZATION
Vol. 4, No. 3, pp. 537–550, August 1994
©1994 Society for Industrial and Applied Mathematics
006

STABILITY RESULTS FOR STOCHASTIC PROGRAMS AND SENSORS, ALLOWING FOR DISCONTINUOUS OBJECTIVE FUNCTIONS*

ZVI ARTSTEIN† AND ROGER J.-B. WETS‡

Abstract. The paper examines the stability of the optimal value and the solutions of stochastic programming problems. Stability is checked with respect to variations in both the problem formulation and the probability distribution that describes the uncertainty. Of particular interest is the case where the payoff functions may be discontinuous. These results are applied to analyze the stability of sensors that model the possibility of making inquiries to improve the probabilistic information available about the uncertain quantities.

Key words. stochastic programming, stability, discontinuous payoff functions, sensors

AMS subject classifications. 90C15, 90C31

1. Introduction.

In this paper we study the stability of the value and solutions of stochastic programs. The stability checked is with respect to perturbations in both the payoff functional and the probability distribution that describes the uncertainty. For the probability distribution we consider perturbations in the sense of the weak convergence of measures. It is natural to check the stability with respect to this convergence as it reflects errors in measurements or in sampling. For the payoff functional we find a new convergence notion for integrands. This convergence could be interpreted as epiconvergence on a function space equipped with a convergence notion related to graph convergence. It allows discontinuities, thus enabling us to deal with a richer class of problems than has been possible so far. We provide examples demonstrating the need for discontinuous payoff functionals, and we show how the convergence introduced in this paper applies. The results are then used to verify the stability of the value of sensors.

We work with an abstract formulation. The stochastic optimization problem considered here is of the form

$$(*) \qquad \underset{x\in X}{\text{maximize}} \int_{\Xi} f(x,\xi)P(d\xi),$$

where X and Ξ are complete separable metric spaces and P is a probability measure on Ξ; the latter space is equipped with its Borel structure. The payoff functional f is a mapping

$$f(x,\xi) : X \times \Xi \longrightarrow [-\infty,\infty] ,$$

and it is assumed to be, at least, measurable in the ξ variable. For a fixed x, the integral is taken in the extended sense, and if not defined, it is set equal to $-\infty$.

---

*Received by the editors July 27, 1992; accepted for publication (in revised form) June 23, 1993. This research was supported by a grant from the U.S.–Israel Binational Science Foundation.

†Incumbent of the Hettie H. Heineman Professorial Chair in Mathematics, Department of Theoretical Mathematics, The Weizmann Institute of Science, Rehovot 76100, Israel (mtarts@weizmann.weizmann.ac.il).

‡Department of Mathematics, University of California Davis, Davis, California 95616 (rjbwets@ucdavis.edu).

As mentioned, the stability of problem (*) is examined with respect to variations both in P and in f. Related questions were examined in the literature from various angles; see, e.g., Kall [9], Wang [20], Robinson and Wets [14], Dupačová [7], [8], Römisch and Schultz [16], [17]. These papers consider variations in P to be in the sense of weak convergence. The payoff functionals in these papers are assumed to be continuous. The analysis of Robinson and Wets [14] assumes, for instance, continuity of the integrands and uses uniform convergence on compacta as the convergence mode. Stability with discontinuous functionals was considered by Langen [11] and by Schultz [18]; we comment on these papers in the body of the text. What the present paper offers is a general framework for checking the continuity of the value with respect to functional convergence of discontinuous payoff functions and the weak convergence of the underlying probabilities. Aspects other than continuity, notably the differentiability of the value, are examined in Kall [9], Dupačová [7], [8], and Römisch and Schultz [16], [17]. The setting of our paper is more general, but we examine only continuity properties.

The paper is organized as follows. In §2 we display two examples where discontinuities occur in the payoff functional. These examples are used in later sections to illustrate and check the abstract results. In §3 we examine semicontinuity of the integration in (*) for a fixed decision x and in §4 the continuity of this integral is studied. These results form a major step in the analysis of the stability of the value that we come to in §5. We also provide in §5 a result on the stability of the solution set and a result on robustness of optimal solutions. The applications of the stability analysis to the examples of §2 are explored in §6. Sensors were introduced in [2] as a tool to evaluate inquiries into the structure of the uncertainty; the application of the stability analysis to the stability of sensors is given in §7 along with a telegraphic introduction to the sensors model.

2. Examples. Discontinuous payoff functions arise in a variety of contexts such as linear stochastic programs with random recourse matrix, stochastic programs with discontinuous recourse costs, and multistage stochastic programs with chance constraints. A particularly rich collection of examples is supplied by stochastic integer programming problems; see, e.g., Laporte, Louveaux, and Mercure [12], Schultz [18], [19], and Klein Haneveld, Stougie, and van der Vlerk [10]. We present here two simple examples with discontinuous payoff functions; they are used in the sequel as illustrations for, and tests of, the abstract results.

Example 2.1. The newsboy problem with mandated backorder. This is a simple inventory problem in the form of the classical newsboy problem as treated, for example, in Richmond [13, Chap. 7].

A newsboy must place an order for a number, say x, of copies. Each copy costs him c dollars, while he sells the newspapers at r dollars apiece. These newspapers are sold until the demand is met. If, however, the demand is greater than x, then the newsboy is obliged to backorder the missing copies. This extra effort costs him K dollars, a cost that does not depend on the number of papers to backorder (say, when the newsboy must send out a truck to collect the backorder).

Thus, for a given demand ξ and decision x, the newsboy's payoff is

$$f(x,\xi) = \begin{cases} r\xi - cx & \text{if} \quad \xi \le x, \\ (r-c)\xi - K & \text{if} \quad \xi > x. \end{cases} \tag{2.1}$$

If the demand ξ is known, then $x = \xi$ would maximize the payoff. But in the situation we analyze, the demand ξ is a random variable whose realization is not known to the

newsboy when the order x is placed. The probability distribution P that governs ξ is, however, available to the newsboy. Thus, if the newsboy wishes to maximize expected payoff, his decision problem is

$$\underset{0\le x<\infty}{\text{maximize}} \int_0^\infty f(x,\xi)P(d\xi) \tag{2.2}$$

with f given in (2.1).

Stability of the problem (2.2) means that the value and solutions do not vary too much when only small perturbations occur in the data. One perturbation that should be considered is that the true probability distribution is not exactly equal to P. Another deviation may be that the fixed backorder cost is not K, but another number, hopefully close to K. The place of the discontinuity, namely, the constraint $\xi \le x$ in (2.1), could also be only an approximation of the true value, as a small number of unsatisfactory customers may be allowed. There may be limits not known in advance on the size of the backorder; then the second line in (2.1) is only an approximation of a term

$$(r-c)\min(\xi, L) - K \quad \text{if} \quad \xi > x$$

with L large, and not known in advance. In all the preceding situations, it is clear what should be meant by "small deviations" in the payoff. The convergence introduced in this paper agrees with this intuitive perception.

Example 2.2. A mixed integer-linear stochastic recourse problem. We exhibit a simplified version of the general two-stage stochastic program with the integer-linear recourse process. (The example was added to the paper after a referee had kindly pointed out to us the paper by Schultz [18].) The problem is as follows (we index the parameters by nought for later use).

$$\underset{-\infty<x<\infty}{\text{maximize}}\ c_0(x) - \int_{-\infty}^{\infty} \Phi_0(|x-\xi|)P_0(d\xi), \tag{2.3}$$

where the function Φ_0 is determined by the optimization procedure

$$\Phi_0(b) = \min\{q_0 y + r_0 z : \alpha_0 y + \beta_0 z = b,\ y \in R_+,\ z \in Z_+\}. \tag{2.4}$$

Here $R_+ = [0,\infty)$ and $Z_+ = \{0,1,2,\ldots\}$; the constants q_0, r_0, α_0, and β_0 are assumed positive.

The interpretation is that the decision x is made first with payoff $c_0(x)$. Then a random quantity ξ is realized and a recourse action is performed by choosing y and z such that $\alpha_0 y + \beta_0 z = |x-\xi|$. The cost in turn is $q_0 y + r_0 z$ and at the recourse stage it should be minimized. The quantity z is thought of as the cheaper one (this would be reflected by the condition $q_0/\alpha_0 > r_0/\beta_0$, but we do not need this to derive the results); in turn, z is restricted to be an integer. As we see in §6, it is this nature of the recourse that generates discontinuities in the integrand $\Phi_0(\cdot)$.

We are interested in the stability of the value of (2.3), with respect to perturbations in the data $c_0(\cdot), q_0, r_0, \alpha_0, \beta_0$, and the underlying distribution P_0.

3. Semicontinuity of the integral. In this section we consider a preliminary, yet a crucial step toward the stability analysis. Let Ξ again be a complete separable metric space endowed with its Borel structure. Consider the integral

$$I(g,P) = \int_\Xi g(\xi)P(d\xi), \tag{3.1}$$

where P is a probability measure on Ξ and $g : \Xi \to [-\infty, \infty]$ is a measurable function. We allow $I(g, P) = +\infty$ or $I(g, P) = -\infty$, and when $I(g, P)$ is still not defined, we set it to be $-\infty$. We are interested in the semicontinuity properties of $I(\cdot, \cdot)$ in its two variables. Specifically, for convergence modes yet to be chosen, we look for conditions on a pair (g_0, P_0) such that $I(g, P)$ is semicontinuous at (g_0, P_0), namely, either lower or upper semicontinuous. Note that the integration in (3.1) is a degenerate case of the optimization problem (*), namely, when the decision space is a singleton.

Langen [11, Thm. 3.3] established semicontinuity results for $I(g, P)$ in the spirit of our result in Theorem 3.5 (we thank a referee for pointing this out). The convergence on sequences of functions g_k used in [11] could be called lower semicontinuous convergence, namely, $\xi_k \to \xi_0$ implies $\liminf\ g_k(\xi_k) \geq g_0(\xi_0)$. When continuity is sought, and both upper and lower semicontinuous convergences are assumed, then one gets the continuous convergence, i.e., $\xi_k \to \xi_0$ implies $g_k(\xi_k) \to g_0(\xi_0)$, which is the convergence used by Robinson and Wets [14, Thm. 2.1]. In particular, in the case of continuity, both [11] and [14] need a continuous limit function $g_0(\cdot)$. Here and in the next section we relax the notion of convergence, allowing, in particular, discontinuous limit integrands $g_0(\cdot)$. We propose a type of graph lower convergence for g_k. For the measures P we use, as in [11] and [14], weak convergence of measures. Both notions are presented now.

We denote by $\mathcal{P}$ the family of probability measures on Ξ and by $\mathcal{G}$ the family of measurable functions $g : \Xi \to [-\infty, \infty]$.

DEFINITION 3.1. (See, e.g., Billingsley [5].) *The sequence P_k of probability measures on Ξ converges weakly to P_0 if for every bounded and continuous function $h : \Xi \to (-\infty, \infty)$, the integrals $I(h, P_k)$ converge to $I(h, P_0)$.*

Since Ξ is a complete separable space, the weak convergence on $\mathcal{P}$ is metrizable. The Prohorov metric induces weak convergence; see [5]. The relevance of the weak convergence to variational problems can be observed through the structure of the Prohorov metric. The Wasserstein metrics used by Römisch and Schultz [16], [17] have a similar structure and could be used as well.

We need the following notation. For $g \in \mathcal{G}$ we denote by $lc\ g$ the lower closure of g, namely, $(lc\ g)(\xi) = \liminf\ g(\eta)$ as $\eta \to \xi$. Similarly, $uc\ g$ denotes the upper closure of g, namely, $(uc\ g)(\xi) = \limsup\ g(\eta)$ as $\eta \to \xi$.

DEFINITION 3.2. *The function g_0 in $\mathcal{G}$ is an episublimit of the sequence g_k if*

$$(lc\ g_0)(\xi) \ \leq\ \liminf\ g_k(\xi_k) \quad \text{as} \quad \xi_k \to \xi\ ,$$

for all $\xi \in \Xi$. The function g_0 in $\mathcal{G}$ is a hyposuplimit of g_k if $-g_0$ is an episublimit of the sequence $-g_k$, or equivalently, if

$$(uc\ g_0)(\xi) \ \geq\ \limsup\ g_k(\xi_k) \quad \textit{as} \quad \xi_k \to \xi \quad \textit{for all } \xi \in \Xi.$$

In the case that g_0 is finite-valued and continuous and g_k continuously converge to g_0, then g_0 is both an episublimit and a hyposuplimit of a sequence g_k. It is easy to see that, generally, if the graphs of the functions g_k converge to the closure of the graph of g_0 in the sense of set-convergence in $\Xi \times [-\infty, \infty]$, then g_0 is both an episublimit and a hyposuplimit of the sequence g_k. Notice that g_0 is an episublimit of g_k if the epigraphs of g_k, i.e., $\{(\xi, r) : r \geq g_k(\xi)\}$, lower converge in the Painlevé–Kuratowski sense to the closure of the epigraph of g_0. The closure operation, however, implies that even if sublimit is replaced by limit (i.e., inequality is replaced by equality in the definition), then the limit is not necessarily unique. For instance, let g be the Dirichlet

function, namely, $g(r) = 1$ for r rational in $[0,1]$, and $g(t) = 0$ otherwise. Then the constant sequence $g, g, \ldots$ has $1 - g$ as both episublimit and hyposuplimit.

We are aiming at continuity results for $I(g,P)$ when $\mathcal{P}$ is endowed with weak convergence, and in $\mathcal{G}$ we consider a convergence based on the notions of episublimit and hyposuplimit. Additional conditions must be posed to guarantee the semicontinuity. One issue that may cause problems is the unboundedness of the integrands. Indeed, even for g fixed, $I(g,P)$ is not semicontinuous with respect to the variable P if g is unbounded. Imposing uniform boundedness on g is too restrictive in the framework of variational problems, as objective functions often assume the value $-\infty$. We follow here Robinson and Wets [14] and Kall [9] and restrict ourselves to equitight (uniformly integrable in the terminology of [14]) families, as follows.

DEFINITION 3.3. *Let $\mathcal{W}$ be a subset of pairs (g,P) in $\mathcal{G} \times \mathcal{P}$. The family $\mathcal{W}$ is equitight if for every $\epsilon > 0$ a compact set K_ϵ, and a bound b_ϵ exist, such that for all $(g,P) \in \mathcal{W}$ the following three conditions hold.*

1. $P(\Xi \backslash K_\epsilon) < \epsilon$;
2. $|g(\xi)| \leq b_\epsilon$ *for P-almost every ξ in K_ϵ*;
3. $\int_{\Xi \backslash K_\epsilon} |g(\xi)| P(d\xi) < \epsilon$.

The preceding definition is a slight generalization of Robinson and Wets [14]; the latter does not require explicitly the boundedness condition 2, but the boundedness holds in [14] and follows from the continuity condition on the g_k and the uniform convergence; also, [14] considers families $\mathcal{W}$ of the form $\mathcal{G}_1 \times \mathcal{P}_1$ with $\mathcal{G}_1 \subset \mathcal{G}$ and $\mathcal{P}_1 \subset \mathcal{P}$. Condition 2 could as well be phrased as $|g(\xi)| \leq b_\epsilon$ if $\xi \in K_\epsilon$; indeed, the values of $g(\cdot)$ on a set of P-measure zero do not affect $I(g,P)$. We choose the almost everywhere version to emphasize the possibility of modelling constraints, i.e., allowing $|g(\xi)| = \infty$.

Equitightness does not suffice to guarantee semicontinuity; one needs in addition some information on the measure of the discontinuity points. To this end we need the following terminology.

Notation 3.4. We say that ξ is a lower continuity point of $h : \Xi \to [-\infty, \infty]$, if $\liminf\ h(\eta) \geq h(\xi)$ as $\eta \to \xi$. The set of points that are not lower continuity points of h is denoted by LDisc h. Likewise, ξ is an upper continuity point of h if it is a lower continuity point of $-h$. The set of points that are not upper continuity points is denoted UDisc h.

The sets LDisc h and UDisc h may not be measurable. When h itself is measurable, these sets are analytic. Indeed, LDisc h is given by

$$\bigcup_{\epsilon>0} \bigcap_{\delta>0} \mathrm{proj}_1\{(\xi,\eta) : d(\xi,\eta) < \delta,\ h(\eta) - h(\xi) > -\epsilon\}, \tag{3.2}$$

where proj_1 is the projection on the first coordinate of (ξ,η), and $d(\cdot,\cdot)$ is the metric on Ξ. The subset of $\Xi \times \Xi$ appearing in (3.2) is then measurable, hence its projection is analytic (see the projection theorem in Castaing and Valadier [6, III.4.23]). Since the intersection and the union in (3.2) can be performed denumerably, the analyticity of the resulting set follows. Being analytic, the set LDisc g for $g \in \mathcal{G}$ belongs to the completion of the Borel field of Ξ with respect to any measure P in $\mathcal{P}$. In particular, the values $P(\text{LDisc } g)$ for $g \in \mathcal{G}$ and $P \in \mathcal{P}$ are well defined. The following is the main result of the present section.

THEOREM 3.5. *Suppose that g_0 is an episublimit of g_k and suppose that P_k converge weakly to P_0 in $\mathcal{P}$. Suppose that $\mathcal{W} = \{(g_k, P_k) : k = 0, 1, 2, \ldots\}$ is equitight, and that $P_0(\text{LDisc } g_0) = 0$. Then* $\liminf\ I(g_k, P_k) \geq I(g_0, P_0)$.

Proof. Let $\epsilon > 0$ be fixed. Let K_ϵ and b_ϵ be provided by the equitightness of $\mathcal{W}$ (Definition 3.3). Since $P_0(\text{LDisc } g_0) = 0$, it follows that for every $\delta > 0$ an open set D_δ exists such that D_δ contains LDisc g_0 and $P_0(D_\delta) < \delta$. We choose δ and D_δ such that $\delta b_\epsilon \leq \epsilon$.

Consider the set $K = K_\epsilon \backslash D_\delta$. This set is compact and contains only lower continuity points of g_0. In particular, g_0 is lower semicontinuous on K. As such, the restriction of g_0 to K is the pointwise limit of a monotone increasing sequence of continuous functions on K. (See Beer [4] for a proof and a useful algorithm for constructing such a sequence.) Therefore a continuous function, say h_0, on K exists such that

$$\int_K |g_0(\xi) - h_0(\xi)| P_0(d\xi) < \epsilon \,. \tag{3.3}$$

We now extend h_0 to the entire space Ξ by letting $h_0(\xi) = g_0(\xi)$ if $\xi \notin K$. Then, since $h_0(\xi) \leq g_0(\xi)$ on K, it follows that each $\xi \in K$ is a lower continuity point of h_0 and h_0 is an episublimit of g_k. We now prove

$$\liminf \; I(g_k, P_k) - I(h_0, P_0) \geq -8\epsilon \,. \tag{3.4}$$

Together with (3.3), and the observation that ϵ is arbitrarily small, (3.4) would complete the proof.

To verify (3.4), we choose a continuous function h_1 defined on an open neighborhood, say Q_1, of K, and such that $h_1(\xi) = h_0(\xi)$ for $\xi \in K$. Such a function h_1 clearly exists. We claim that an open neighborhood Q of K can be found, such that $Q \subset Q_1$ and such that the following three conditions are satisfied.

(i) $\int_{Q \backslash K} (|h_0(\xi)| + |h_1(\xi)|) P_0(d\xi) < \epsilon$.

(ii) For some k_0, $g_k(\xi) - h_1(\xi) \geq -\epsilon$ for $\xi \in Q$ and $k \geq k_0$.

(iii) $P_0(\partial Q) = 0$ (where ∂Q denotes the boundary of Q).

To verify the existence of such a Q we use a compactness argument as follows. The estimate for the integral in (i) is satisfied for some neighborhood of K since $P_0(K) = 0$; therefore $P_0(Q' \backslash K)$ converge to zero as $Q' \backslash K$ gets smaller. To establish the estimates in (ii) and (iii), consider first a point ξ_0 in K and establish (ii) and (iii) for a neighborhood Q_{ξ_0} of ξ_0. To find a neighborhood of ξ_0 for which (ii) holds, notice that the condition $h_1(\xi_0) \leq (lc\ g_0)(\xi_0)$ and the episublimit property imply that there exists a neighborhood V_{ξ_0} of ξ_0 and a k_0 such that $g_k(\xi') \geq h_1(\xi_0) - \epsilon/2$ if $k \geq k_0$ and $\xi' \in V_{\xi_0}$. On the other hand, the continuity of $h_1(\cdot)$ in a neighborhood of K implies that a neighborhood V'_{ξ_0} of ξ_0 exists such that $h_1(\xi_0) \geq h_1(\xi') - \epsilon/2$ for all $\xi' \in V'_{\xi_0}$. Then for $\xi' \in V_{\xi_0} \cap V'_{\xi_0}$, the inequality $g_k(\xi') \geq h_1(\xi') - \epsilon$ holds for $k \geq k_0$, hence $V_{\xi_0} \cap V'_{\xi_0}$ is a neighborhood of ξ_0 for which (ii) is satisfied. A standard argument could produce a neighborhood Q_{ξ_0} of ξ_0 which is a subset of $V_{\xi_0} \cap V'_{\xi_0}$ and such that $P(\partial Q_{\xi_0}) = 0$. For this Q_{ξ_0} both (ii) and (iii) are satisfied. The compactness of K implies that the union of a finite number of such Q_{ξ_0} with $\xi_0 \in K$ covers K. This union is a neighborhood of K for which (ii) and (iii) are satisfied and its intersection with the neighborhood satisfying (i) is the desired set Q.

The conditions $P_k \to P_0$, and $P_0(\partial Q) = 0$, and the continuity of h_1 on Q, imply that

$$\lim \left| \int_Q h_1(\xi) P_k(d\xi) - \int_Q h_1(\xi) P_0(d\xi) \right| = 0 \,. \tag{3.5}$$

Therefore, by (i) and (ii),

$$\liminf \left(\int_Q g_k(\xi) P_k(d\xi) - \int_Q h_0(\xi) P_0(d\xi) \right) \geq -2\epsilon \,. \tag{3.6}$$

This is not quite what is required in (3.4) as the integration in the latter is on the entire space Ξ. We note, however, that the equitightness implies that

$$\int_{\Xi \setminus K_\epsilon} |g_k(\xi)| P_k(d\xi) < \epsilon \tag{3.7}$$

for all $k = 0, 1, 2, \ldots$ (and recall that $h_0 = g_0$ on $\Xi \setminus K_\epsilon$). Also, since $K_\epsilon \setminus Q$ is a compact set included in D_δ, it follows that $P_0(K_\epsilon \setminus Q) \leq \delta$, and, since $P_k \to P_0$, it follows that

$$\limsup \int_{K_\epsilon \setminus Q} |g_k(\xi)| P_k(d\xi) \leq \delta b_\epsilon \leq \epsilon \,. \tag{3.8}$$

Combining (3.6), (3.7), (3.8), and noting that $Q \cup (\Xi \setminus K_\epsilon) \cup (K_\epsilon \setminus Q) = \Xi$, the estimate in (3.4) is verified. This completes the proof. □

The following is the upper semicontinuity result, analogous to the previous one; it follows immediately from the definition of hyposuplimit and the lower semicontinuity result.

THEOREM 3.6. *Suppose g_0 is a hyposuplimit of g_k, and that P_k converge weakly to P_0. Suppose that the family $\mathcal{W} = \{(g_k, P_k) : k = 0, 1, \ldots\}$ is equitight and $P_0(\text{UDisc } g_0) = 0$. Then* $\limsup I(g_k, p_k) \leq I(g_0, P_0)$.

It is clear that the assumptions $P_0(\text{LDisc } g_0) = 0$ and $P_0(\text{UDisc } g_0) = 0$ cannot be removed from the previous results. Such an assumption is not used in Langen [11, Thm. 3.3] as the semicontinuous convergence used in [11] (see the early part of this section) makes it irrelevant where the discontinuities occur.

4. Continuity of the integral. We use the two theorems of the previous section to derive conditions guaranteeing the continuity of $I(g, P)$. To this end we need the following terminology.

Notation 4.1. For $h : \Xi \to [-\infty, \infty]$ we denote by Disc h the set of points ξ in Ξ at which h is discontinuous.

Note that Disc h is the union of LDisc h and UDisc h. The set Disc h is Borel even if h itself is not measurable. Indeed, for a bounded h the set Disc h is a denumerable union of closed sets $\text{Disc}_\epsilon\, h$ consisting of the points ξ in Ξ at which the discontinuity is at least ϵ, i.e., $\limsup |h(\eta) - h(\zeta)| \geq \epsilon$ as η and ζ converge to ξ. For unbounded h we notice that Disc h is preserved under the transformation $h \to h(1 + |h|)^{-1}$ (with $\infty \cdot \infty^{-1} = 1$) that transforms h into a bounded mapping.

THEOREM 4.2. *Suppose that g_0 is both an episublimit and a hyposuplimit of the sequence g_k and that P_k converges weakly to P_0. Suppose that the family $\mathcal{W} = \{(g_k, P_k) : k = 0, 1, \ldots\}$ is equitight and $P_0(\text{Disc } g_0) = 0$. Then $I(g_k, P_k)$ converge to $I(g_0, P_0)$.*

Proof. The conditions of both Theorems 3.5 and 3.6 hold, and the two semicontinuity conclusions imply the continuity. □

Remark 4.3. A referee pointed out to us that using Rubin's Theorem, Theorem 4.2 has an alternative direct proof (see Billingsley [5, Thm. 5.5]). A proper change of variables in our problem reduces the convergence to the case of Rubin's result. Our approach relies on the semicontinuity of the preceding section. This semicontinuity is

interesting for its own sake, and our proof yields also the estimate in Proposition 4.5. In turn, it might be interesting to interpret our semicontinuity result in the framework of Rubin's Theorem.

Remark 4.4. As we mentioned already, the continuity result improves available results; e.g., Robinson and Wets [14] where all integrands are assumed continuous (hence $P_0(\text{Disc } g_0) = 0$ since Disc g_0 is empty) and the convergence $g_k \to g_0$ is uniform on compact sets. Our setting is more relaxed, but notice that under the conditions of Theorem 4.2, the functions g_k converge to g_0 uniformly on compact subsets of the continuity points of g_0. The assumption $P_0(\text{Disc } g_0) = 0$ that we add is natural in the presence of discontinuities. (It is used in Billingsley [5, Chap. 5] in the analysis of weak convergence, which in turn is used in Schultz [18] in his study of stability in stochastic programs; it was also used in [1] in the context of variational limits.) If this assumption is dropped, the convergence may fail even for g_0 fixed. To get an example, let $g_0(\xi) = 0$ for $0 \le \xi < 1$ and $g_0(\xi) = 1$ for $1 \le \xi \le 2$, let P_0 have an atom at $\xi = 1$, and let P_k have an atom at $\xi = 1 - k^{-1}$. Similarly, the continuity may then fail for P_0 fixed. For instance, let $g_0(\xi) = 0$ for $0 \le \xi \le 1$, $g_0(\xi) = 1$ for $1 < \xi \le 2$ and P_0 as before and $g_k(\xi) = \min(\xi^k, 1)$.

Without the assumption $P_0(\text{Disc } g_0) = 0$, the continuity of the integral may fail. Our method of proof, however, provides an estimate for the discontinuity that may result. We present here a result along this direction in the case where the integrands are bounded; analogous results can be derived for the equitight case.

PROPOSITION 4.5. *Suppose that g_0 is both an episublimit and a hyposuplimit of g_k and $P_k \to P_0$. Suppose that g_k are uniformly bounded by the same bound β. Let $\Delta = P_0(\text{Disc } g_0)$. Then*

$$\limsup |I(g_k, P_k) - I(g_0, P_0)| \le 2\beta\Delta \ .$$

Proof. We follow the proof of the lower semicontinuity in Theorem 3.5, but choose D_δ to be such such that $P_0(D_\delta) \le \Delta + \delta$. This affects the analysis when (3.8) is derived and $\delta\beta_\epsilon$ there should be replaced by $(\delta + \Delta)\beta$. This change is carried back to (3.4) and the effect is that -8ϵ is replaced by $-2\Delta\beta - 8\epsilon$, and since ϵ is arbitrarily small, we are left with $-2\Delta\beta$. The analogous argument for the upper semicontinuity completes the proof. □

5. Stability analysis. In this section we use the semicontinuity and the continuity results of the previous sections to derive stability results for the stochastic optimization problem (∗). We consider a sequence (f_k, P_k) of pairs of integrands and probability measures. We find conditions under which the values of the corresponding stochastic problems converge to the value associated with the pair (f_0, P_0). To this end let

$$\text{val}(f_k, P_k) = \operatorname*{supremum}_{x \in X} \int_\Xi f_k(x, \xi) P_k(d\xi), \tag{5.1}$$

where the integral is set to be equal to $-\infty$ if not defined.

As a preliminary step toward the stability, we consider the convergence of the functionals γ_k defined by

$$\gamma_k(x) = \int_\Xi f_k(x, \xi) P_k(d\xi) \ . \tag{5.2}$$

Notice that $\text{val}(f_k, P_k)$ is the supremum over $x \in X$ of $\gamma_k(x)$.

A major tool in the analysis of stability of minimization problems is the concept of epiconvergence of the cost functional that becomes hypoconvergence in the case of maximization problems; for a detailed account, see Attouch [3] or Rockafellar and Wets [15]. Recall that the sequence $\gamma_k : X \to [-\infty, \infty]$ hypoconverges to γ_0 if at every point x, lim sup $\gamma_k(x_k) \leq \gamma_0(x_0)$ when $x_k \to x_0$ and there exists at least one sequence $y_k \to x_0$ such that lim $\gamma_k(y_k) = \gamma_0(x_0)$. (Hypoconvergence can be characterized in terms of the set-convergence of the hypographs; see [15].)

We now state the main connections between convergence properties of the pairs (f_k, P_k) and the associated upper semicontinuity properties and hypoconvergence for the functions γ_k.

PROPOSITION 5.1. *Suppose that whenever $x_k \to x_0$ in X, the function $f(x_0, \cdot)$ is a hyposuplimit of $f(x_k, \cdot)$, and $\mathcal{W} = \{(f(x_k, \cdot), P) : k = 0, 1, \ldots\}$ is equitight. Suppose also that $P(\text{UDisc } f(x_0, \cdot)) = 0$ for all x_0. Then γ is upper semicontinuous.*

Proof. The assumptions of Theorem 3.6 hold for $g_k(\cdot) = f(x_k, \cdot)$ and $P_k = P$ for all k. The conclusion of Theorem 3.6 amounts to the upper semicontinuity of γ. □

THEOREM 5.2. *Suppose that P_k converge to P_0 and that whenever $x_k \to x_0$, the function $f_0(x_0, \cdot)$ is a hyposuplimit of $f_k(x_k, \cdot)$ and that for each x_0 there is at least one sequence $y_k \to x_0$ such that $f_0(x_0, \cdot)$ is an episublimit of $f_k(y_k, \cdot)$. Suppose that whenever $x_k \to x_0$, the set $\mathcal{W} = \{(f_k(x_k, \cdot), P_k) : k = 0, 1, \ldots\}$ is equitight and that $P_0(\text{Disc } f_0(x_0, \cdot)) = 0$ for all x_0. Then γ_k hypoconverge to γ_0.*

Proof. By Theorem 3.6, lim sup $\gamma_k(x_k) \leq \gamma_0(x_0)$ whenever $x_k \to x_0$, and by Theorem 3.5, limin f $\gamma_k(y_k) \geq \gamma_0(x_0)$ for the specific sequence y_k guaranteed by the conditions. This completes the proof. □

Notice the structure in the previous result. The conditions resemble a kind of hypoconvergence for the functional $f(x, \cdot)$, but with respect to the hyposuplimit and the episublimit notions for functions defined on Ξ, rather than with respect to convergence of real numbers. The additional conditions (equitightness and the measure zero condition) guarantee that this hypolimit notion for function-valued functionals $f_k(x, \cdot)$, is translated into the standard hypolimit of the functions γ_k.

Once the upper semicontinuity and the hypoconvergence of γ_k are established, existence results and the stability of the value follow in a direct manner. We display here several such results. A more elaborate discussion, which could be applied here, can be found in Rockafellar and Wets [15].

THEOREM 5.3. *Under the conditions of Proposition* 5.1, *if X is compact, then a solution to (*) exists.*

Proof. An upper semicontinuous function on a compact set attains its maximum, hence γ attains its maximum, and the point x at which the maximum is attained is an optimal solution. □

THEOREM 5.4. *Under the conditions of Theorem* 5.2, *if X is compact, then* val(f_k, P_k) *converges to* val(f_0, P_0).

Proof. Let x_k be such that $\text{val}(f_k, P_k) - \gamma_k(x_k) \leq k^{-1}$. Compactness implies that x_k has a converging subsequence, say the whole sequence converges to x_0. The conditions imply that $f_0(x_0, \cdot)$ is a hyposuplimit of $f_k(x_k, \cdot)$. By Theorem 3.6, lim sup $\gamma_k(x_k)$ $\leq \gamma_0(x_0)$. Since $\gamma_0(x_0) \leq \text{val}(f_0, P_0)$, the choice of x_k concludes the first part of the result. For the second part note that the conditions of Theorem 3.5 are fulfilled for $g_k(\cdot) = f_k(y_k, \cdot)$ and P_k, with $y_k \to x_0$ and x_0 such that $\text{val}(f_0, P_0) - \gamma_0(x_0) \leq \epsilon$ and ϵ arbitrarily small. Then $\gamma_k(y_k) \to \gamma_0(x_0)$, which proves that lim inf val$(f_k, P_k) \geq$ val(f_0, P_0), and together with the first part, the proof is complete. □

We present a result concerning the behavior of the solution set of (*). Denote by $\sigma(f_k, P_k)$ the set of x such that $\gamma_k(x) = \text{val}(f_k, P_k)$, namely, $\sigma(f_k, P_k) = \text{argmax } \gamma_k$.

THEOREM 5.5. *Suppose that the conditions of Proposition* 5.1 *hold for each* f_k, $k = 0, 1, \ldots$. *Suppose also that the conditions of Theorem* 5.2 *hold. If* X *is compact, then* $\limsup\ \sigma(f_k, P_k)$ *is included in* $\sigma(f_0, P_0)$.

Proof. From Proposition 5.1, γ_k is upper semicontinuous and from Theorem 5.2 γ_k hypoconverge to γ_0. The semicontinuity of the set-valued mapping argmax γ_k then follows in a standard way. □

We conclude our stability analysis with the question of robustness, namely, suppose that an optimal solution for the data (f_0, P_0) is applied to the problem with the data (f_k, P_k). Is the resulting error small if k is large?

THEOREM 5.6. *Suppose that the conditions of Theorem* 5.2 *hold and let* x^* *be an optimal solution for the data* (f_0, P_0). *Suppose also that* $f_0(x^*, \cdot)$ *is an episublimit of* $f_k(x^*, \cdot)$. *Then* x^* *is an approximate solution for the data* (f_k, P_k) *for* k *large, namely,* $\mathrm{val}(f_k, P_k) - \gamma_k(x^*)$ *converge to zero as* $k \to \infty$.

Proof. The continuity result of Theorem 5.2 together with x^* being an optimal solution imply that

$$\mathrm{val}(f_k, P_k) - \gamma_0(x^*) \longrightarrow 0\ ,$$

and since $\gamma_k(x^*) \leq \mathrm{val}(f_k, P_k)$, it follows that

$$\limsup\ \gamma_k(x^*) - \gamma_0(x^*) \leq 0\ .$$

Applying Theorem 3.5 with the present conditions implies

$$\liminf\ \gamma_k(x^*) - \gamma_0(x^*) \geq 0\ .$$

Hence $\gamma_k(x^*)$ converges to $\gamma_0(x^*)$, and together with the convergence of $\mathrm{val}(f_k, P_k)$ to $\gamma_0(x^*)$, the proof is complete. □

We note that a bound for the discontinuity gap, as presented in Proposition 4.5 for integrals, is valid also in the case of the optimization problems.

6. On the examples. In this section we examine the examples introduced in §2 in light of the stability results of the previous section.

Consider first Example 2.1. For the sake of the illustration, we consider the cost functional

$$f_0(x, \xi) = \begin{cases} r\xi - cx & \text{if } \xi \leq x, \\ (r-c)\xi - K_0 & \text{if } \xi > x, \end{cases} \tag{6.1}$$

where ξ is governed by the exponential distribution with parameter λ_0, namely, $P_0(d\xi) = \lambda_0 e^{-\lambda_0 \xi} d\xi$. A straightforward computation shows then that the maximization problem

$$\underset{0 \leq x < \infty}{\text{maximize}} \int_0^\infty f_0(x, \xi) \lambda_0 e^{-\lambda_0 \xi} d\xi \tag{6.2}$$

translates into the problem

$$\underset{0 \leq x < \infty}{\text{maximize}} \left(\frac{r}{\lambda_0} - cx - \left(K_0 + \frac{c}{\lambda_0} \right) e^{-\lambda_0 x} \right)\ . \tag{6.3}$$

Solving (6.3) is a simple exercise (note that the payoff function is concave on $[0, \infty)$), and the optimal solution is

$$x^* = \frac{1}{\lambda_0} \log \left(1 + \frac{\lambda_0 K_0}{c} \right). \tag{6.4}$$

We consider now the possibility that the payoff functional (6.1) is only an approximation of the true payoff functional. In view of the discussion in §2, consider the payoff functionals

$$f_k(x,\xi) = \begin{cases} r\xi - cx & \text{if } \xi \le x + \epsilon_k(x), \\ (r-c)\min(\xi, L_k) - K_k & \text{if } \xi > x + \epsilon_k(x) . \end{cases} \tag{6.5}$$

Recall that $\epsilon_k(x)$ reflects the possibility that a number of customers may remain unsatisfied; the deviation K_k from K_0 reflects an error in the cost of the backorder, and L_k reflects the possibility that the backorder is limited. We also wish to consider the possibility that the parameter λ_0 of the underlying probability measure in (6.2) is only an approximation of the true exponent λ_k of the probability distribution. Thus, we consider (6.2) an approximation of

$$\underset{0\le x<\infty}{\text{maximize}} \int_0^\infty f_k(x,\xi)\lambda_k e^{-\lambda_k \xi} d\xi . \tag{6.6}$$

THEOREM 6.1. *Suppose that $\lambda_k \to \lambda_0$, $K_k \to K_0$, $L_k \to \infty$, and $\epsilon_k(x)$ converge uniformly to zero. Then the value of* (6.6) *converges to the value of* (6.2). *Furthermore, if the optimal solution x^* of* (6.2), *given in* (6.4), *is applied to the problem* (6.6) *with k sufficiently large, then the resulting payoff is a good approximation of the optimal payoff.*

Proof. The result follows from Theorems 5.4 and 5.6 once the conditions of these results are verified. We comment on these conditions. The condition $P_0(\text{Disc } f_0(x,\cdot)) = 0$ holds since the discontinuity of the payoff function (6.1) occurs on a set of Lebesgue measure zero (at one point actually) and P_0 is absolutely continuous. The equitightness follows since $|f_0(x,\cdot)|$ grows linearly in ξ, while $\lambda_k e^{-\lambda_k \xi}$ decays exponentially, uniformly as $\lambda_k \to \lambda_0$. The graph-type convergence, i.e., episublimit and hyposuplimit are easy to verify under the convergence conditions. Finally, it is easy to verify that the search for optimal solutions can be restricted to a compact set of possible orders. Checking all that completes the proof. □

We examine now Example 2.2. We are interested in the stability of the value for a sequence of problems determined by the data $D_k = (c_k(x), q_k, r_k, \alpha_k, \beta_k, P_k)$ for $k = 0, 1, 2, \ldots$. Denote the value of the problem (2.3) with D_k instead of D_0 by v_k. We find conditions guaranteeing that $v_k \to v_0$.

Before addressing the stability problem, we wish to identify the points of discontinuity of $\Phi_k(|x-\xi|)$; the discontinuities are caused by the structure of the recourse stage. The problem we choose is simple enough to allow the computation of the discontinuities (see Remark 6.3). Indeed, for b fixed, only a finite number of $z \in Z_+$ participate in the minimization (2.4), namely, those with $\beta_0 z \le b$. Therefore, for x fixed, the only possible discontinuities are at $|x-\xi| = \beta_k j$ for $j = 0, 1, \ldots$.

We identify the following two ensembles. Let Δ denote the family of probability measures P on $(-\infty, \infty)$ such that $\int_{-\infty}^{\infty} |\xi| P(d\xi) \le \delta_0$ with δ_0 a prescribed constant. Let Γ denote the family of continuous functions $c(x) : (-\infty, \infty) \to R$, such that $c(x) \le \kappa_0(x)$ with $\kappa_0(\cdot)$ a prescribed function satisfying $\kappa_0(x) \to -\infty$ as $|x| \to \infty$.

THEOREM 6.2. *Suppose that $q_k \to q_0$, $r_k \to r_0$, $\alpha_k \to \alpha_0$, and $\beta_k \to \beta_0$. Suppose that all $c_k(\cdot)$ belong to Γ and that $c_k(x) \to c_0(x)$ uniformly on compact sets. Suppose that all P_k belong to Δ and that $P_k \to P_0$ weakly. Finally suppose that P_0 is atomless. Then $v_k \to v_0$. Furthermore, if x^* is an optimal solution of* (2.3), *and x^* is used as the decision for the data D_k, then k large implies that the resulting cost is close to the optimal one.*

Proof. The claims follow from Theorems 5.4 and 5.6 once the conditions are verified. We comment on these conditions. The definitions of Δ and Γ imply that the search for an optimal x can be limited to a compact interval. Then the required tightness of P_k follows from the linear bound on the growth of the value in the recourse stage. The functional convergence of $\Phi_k(|x-\xi|)$ is transparent at all continuity points. The structure of discontinuities that we displayed earlier shows that the required functional convergence holds at all points. Finally, for P_0 atomless, the requirement $P_0(\text{Disc } \Phi_k(|x-\cdot|))$ holds since the set of discontinuities is discrete. This completes the proof. □

Remark 6.3. The preceding example is a much simplified version of the general mixed integer-linear two-stage stochastic program analyzed by Schultz [18]. We thank a referee for pointing out this reference. We then added the example to show how the considerations of Schultz [18] could be incorporated into our framework. Indeed, Schultz verifies the continuity of the value as $P_k \to P_0$ (and P_k belong to a family of a type similar to Δ), when P_0 is absolutely continuous with respect to the Lebesgue measure. This is indeed the right general condition. As in the case with several dimensions analyzed in [18], one can only guarantee that the discontinuities of $\Phi(|x-\cdot|)$ occur in a denumerable number of hyperplanes; in our case these are points, so a condition of being atomless suffices.

7. Stability of sensors. Sensors were introduced in [2] as a tool to evaluate the worth of an inquiry into the structure of uncertainty. We recall here the basic idea and definition of the concept (the interested reader can check [2] for the theory and examples). We then verify the stability of sensors with possibly discontinuous objective functions. In fact, given the preceding results in the paper, the stability of sensors follow quite easily.

Consider the maximization problem (*). In many real situations, the decision-maker has an option of acquiring information about the random event ξ before choosing x. This information may not determine ξ completely; rather, a more accurate probability may be obtained. For instance, the newsboy in Example 2.1 may conduct a market research, say by sampling the prospective buyers. The given probability P may be improved by conditioning on the result, say r, of the market research. Let us denote the emerging probability by P_r. Before deciding on the market research (which may involve a cost), the decisionmaker knows what possible outcomes P_r may emerge and he may also know the probability that governs the distribution of the results r; hence he knows the distribution that governs the possible probabilities P_r. This probability on probabilities we call a sensor. Thus a sensor reflects the possible outcome of a market research before such research is conducted. But note that different market research procedures may lead to different sensors and the decisionmaker may need to decide which one to follow. To this end we introduce the following.

DEFINITION 7.1. A sensor, say S, is a probability distribution on $\mathcal{P}$. The value of the sensor S, when applied to the problem (*), is

$$\text{val}(f,S) = \int_{\mathcal{P}} \text{val}(f,P)S(dP)\ . \tag{7.1}$$

The rationale of defining the value of the sensor S as the expectation in (7.1) is as follows. The value of a fixed probability P, given f, is $\text{val}(f,P)$. The occurrence of P is governed probabilistically by the sensor S, hence expectation of $\text{val}(f,P)$ with respect to the measure S is the value of S to a decisionmaker who wishes to maximize expected payoff.

Applying a sensor S may be subject to errors and deviations generated, e.g., by errors in the sampling that determines S. We are interested in the stability of $v(f,S)$ with respect to variations both in f and in S. For convergence in the space of sensors, we take the weak convergence of measures. In [2], continuity of the value with respect to S was established for a payoff function f fixed and continuous. Here we use the analysis of the previous section and establish continuity with respect to the pair (f,S) with payoff functionals possibly discontinuous.

We denote by supp S the support of S, namely, the smallest closed set M of probability measures in $\mathcal{P}$ such that $S(M)=1$.

THEOREM 7.2. *Suppose that the functions $f_k(x,\xi)$ are uniformly bounded, $k=0,1,2,\ldots$. Suppose that whenever $x_k \to x_0$, the function $f_0(x_0,\cdot)$ is a hyposuplimit of $f_k(x_k,\cdot)$, and that for every x_0 a sequence $y_k \to x_0$ exists such that $f_0(x_0,\cdot)$ is an episublimit of $f_k(y_k,\cdot)$. Suppose that S_0 is a sensor such that $P(\text{Disc } f_0(x_0,\cdot))=0$ for every x_0 and every $P \in \text{supp } S_0$. Then, if S_k converge to S_0, the values $\text{val}(f_k,S_k)$ converge to $\text{val}(f_0,S_0)$.*

Proof. Denote $v_k(P)=\text{val}(f_k,P)$, namely, the value of $(*)$ with f_k the payoff function and P the probability distribution. A particular case of Theorem 5.2 implies that $v_0(P)$ is continuous on supp S_0. The general case of Theorem 5.2 implies that $v_k(P_k)$ converges to $v_0(P_0)$ if P_k converges weakly to P_0. With these properties, the Lebesgue dominated convergence theorem implies

$$\int v_k(P)S_k(dP) \quad \text{converge to} \quad \int v_0(P)S_0(dP)\ .$$

In view of (7.1), this convergence is the desired conclusion and the proof is now complete. □

The preceding result is stated for f_k uniformly bounded, which makes the equitightness condition trivial. Unbounded functionals can be incorporated with the proper conditions of equitightness for $(f_k(x,\cdot),P_k)$ whenever $P_k \to P$ and $P \in \text{supp } S_0$. We leave out the details.

REFERENCES

[1] Z. ARTSTEIN, *Chattering variational limits of control systems*, Forum Mathematicum, 5 (1993), pp. 369–403.

[2] Z. ARTSTEIN AND R.J.-B. WETS, *Sensors and information in optimization under stochastic uncertainty*, Math. Oper. Res., 18 (1993), pp. 523–547.

[3] H. ATTOUCH, *Variational Convergence for Functions and Operators*, Pitman, Boston, 1984.

[4] G. BEER, *A geometric algorithm for approximating semicontinuous functions*, J. Approx. Theory, 49 (1987), pp. 31–40.

[5] P. BILLINGSLEY, *Convergence of Probability Measures*, Wiley, New York, 1968.

[6] C. CASTAING AND M. VALADIER, *Convex Analysis and Measurable Multifunctions*, Lecture Notes in Math. 580, Springer-Verlag, Berlin, 1977.

[7] J. DUPAČOVÁ, *Stability and sensitivity analysis for stochastic programming*, Ann. Oper. Res., 27 (1990), pp. 115–147.

[8] J. DUPAČOVÁ, *On statistical sensitivity analysis in stochastic programming*, Ann. Oper. Res., 30 (1991), pp. 199–214.

[9] P. KALL, *On approximation and stability in stochastic programming*, in Parametric Optimization and Related Problems, J. Guddat et al. eds., Mathematische Research Band 35, Akademie-Verlag, Berlin 1987, pp. 387–407.

[10] W. K. KLEIN HANEVELD, L. STOUGIE, AND M. H. VAN DER VLERK, *Stochastic integer programming with simple recourse*, Research Memorandum No. 455, Institute of Economic Research, Rijksuniversiteit Groningen, 1991.

[11] H. J. LANGEN, *Convergence of dynamic programming models*, Math. Oper. Res., 6 (1981), pp. 493–512.

[12] G. LAPORTE, F. LOUVEAUX, AND H. MERCURE, *The vehicle routing problem with stochastic travel times*, Technical Report No. 96, Faculté Notre-Dame de la Paix, 1989.

[13] S. B. RICHMOND, *Operations Research for Management Decisions*, Ronald Press Co., New York, 1968.

[14] S. M. ROBINSON AND R. J.-B. WETS, *Stability in two-stage stochastic programming*, SIAM J. Control Optim., 25 (1987), pp. 1407–1416.

[15] R. T. ROCKAFELLAR AND R. J.-B. WETS, *Variational analysis, an introduction*, in Multifunctions and Integrands, Stochastic Analysis and Optimization, G. Salinetti, ed., Lecture Notes in Math. 1091, Springer-Verlag, Berlin, 1984, pp.1–54.

[16] W. RÖMISCH AND R. SCHULTZ, *Distribution sensitivity in stochastic programming*, Math. Programming, 50 (1991), pp. 197–226.

[17] ———, *Stability analysis for stochastic programs*, Ann. Oper. Res., 30 (1991), pp. 241–266.

[18] R. SCHULTZ, *Continuity and stabilization in two-stage stochastic integer programming*, in Stochastic Optimization, Numerical Methods and Technical Applications, K. Marti, ed., Lecture Notes in Economics and Math. Systems 379, Springer-Verlag, Berlin 1992, pp. 81–92.

[19] ———, *Continuity properties of expectation functions in stochastic integer programming*, Math. Oper. Res., 18 (1993), pp. 578–589.

[20] J. WANG, *Distribution sensitivity analysis for stochastic programs with complete recourse*, Math. Programming, 31 (1985), pp. 286–297.

SIAM J. OPTIMIZATION
Vol. 4, No. 3, pp. 551–572, August 1994
©1994 Society for Industrial and Applied Mathematics
007

PARTIAL PROXIMAL MINIMIZATION ALGORITHMS FOR CONVEX PROGRAMMING*

DIMITRI P. BERTSEKAS† AND PAUL TSENG‡

Abstract. An extension of the proximal minimization algorithm is considered where only some of the minimization variables appear in the quadratic proximal term. The resulting iterates are interpreted in terms of the iterates of the standard algorithm, and a uniform descent property is shown that holds independently of the proximal terms used. This property is used to give simple convergence proofs of parallel algorithms where multiple processors simultaneously execute proximal iterations using different partial proximal terms. It is also shown that partial proximal minimization algorithms are dual to multiplier methods with partial elimination of constraints, and a relation is established between parallel proximal minimization algorithms and parallel constraint distribution algorithms.

Key words. proximal minimization, duality, augmented Lagrangian, mathematical programming

AMS subject classifications. primary 90C47; secondary 90C05

1. Introduction.

Let us consider the proximal minimization algorithm defined by

$$x^{k+1} = \arg\min_{x \in \Re^n} \left\{ f(x) + \frac{1}{2c}\|x - x^k\|^2 \right\}. \tag{1}$$

Here c is a positive constant, $\|\cdot\|$ denotes the standard Euclidean norm on $\Re^n$, and f is a closed proper convex function in $\Re^n$ [Roc70], that is, an extended-real-valued, lower semicontinuous convex function on $\Re^n$, which is not identically $+\infty$. It is well known that, starting from an arbitrary $x^0 \in \Re^n$, the sequence $\{x^k\}$ converges to a minimizer of f if there exists at least one minimizer and diverges otherwise.

The algorithm, originally proposed by Martinet [Mar70], [Mar72] and further refined and extended by Rockafellar [Roc76], is useful for "regularizing" the minimization of f through the addition of the strongly convex term $\|x - x^k\|^2$. The algorithm is also useful in a dual context, where f is the dual function of a constrained nonlinear programming problem and x is a vector of Lagrange multipliers. Then, by using the Fenchel duality theorem, the proximal iteration (1) can be interpreted as an augmented Lagrangian iteration, that is, a minimization of a quadratic augmented Lagrangian function associated with the primal problem followed by a Lagrange multiplier update. This interpretation, first given by Rockafellar [Roc73], can be found in several sources, e.g., [Ber82], [BeT89].

In this paper we focus on a variation where the vector x is partitioned in two subvectors x_1 and x_2

$$x = (x_1, x_2)$$

*Received by the editors November 26, 1991; accepted for publication (in revised form) April 12, 1993. This research was supported by National Science Foundation grants DDM-8903385 and CCR-9103804, and by Army Research Office grant DAAL03-86-K-0171.

†Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139 (dimitrib@mit.edu).

‡Department of Mathematics, University of Washington, Seattle, Washington 98195.

with $x_1 \in \Re^{n_1}$, $x_2 \in \Re^{n_2}$, $n_1 + n_2 = n$, and the proximal term $\|x - x^k\|^2$ is replaced by the portion involving only the subvector x_1, that is,

$$(x_1^{k+1}, x_2^{k+1}) \in \arg \min_{(x_1,x_2)\in\Re^n} \left\{ f(x_1, x_2) + \frac{1}{2c}\|x_1 - x_1^k\|^2 \right\}. \tag{2}$$

We call this the *partial proximal minimization algorithm*, but hasten to observe that it can be viewed as a special case of the ordinary algorithm (1) with $f(x_1, x_2)$ replaced by

$$f_1(x_1) = \min_{x_2\in\Re^{n_2}} f(x_1, x_2),$$

assuming that the minimum of f above is attained for all $x_2 \in \Re^{n_2}$. Thus, the sequence $\{x_1^k\}$ is in effect generated by applying the ordinary algorithm to the function f_1, while x_2^k is obtained by minimizing $f(x_1^k, x_2)$ with respect to x_2, after x_1^k has been computed. It follows that the convergence properties of the partial algorithm can be inferred from those of the ordinary one; in fact this has been demonstrated by Ha [Ha90]. The partial algorithm, however, allows a choice between several partial proximal terms, and also allows the simultaneous use of several different proximal terms in a parallel computing context. When such possibilities are considered, the theory of the ordinary algorithm is not directly applicable and a new analysis is needed. Our main purpose in this paper is to provide such an analysis.

Our interest in the partial algorithm stems from recent work by Ferris and Mangasarian [FeM91] and Ferris [Fer91a] on parallel constraint distribution. Parallel constraint distribution is an augmented Lagrangian type algorithm for convex programming whereby at each iteration the constraints are partitioned into subsets and, for each subset, an augmented Lagrangian subproblem in which constraints not of the subset appear in the augmented Lagrangian, is solved; the multipliers obtained from each of the subproblems are then combined in some simple fashion to yield a new set of multipliers. This algorithm has the advantage that each subproblem has fewer constraints than the original problem and the subproblems can be solved in parallel. Numerical test results reported in [FeM91] and [Fer91a,b] indicate that the algorithm is quite promising for practical computation, especially when implemented in parallel. One of our purposes in this paper is to show that parallel constraint distribution is closely related to proximal minimization and particularly to a parallel implementation of the partial proximal iteration (2) (see §5).

A central observation of the present paper is that the partial proximal iteration (2) can be decomposed into the following sequence of two steps as shown in Fig. 1.

(a) A (block) coordinate descent iteration for the function F_c defined by

$$F_c(x) = \min_{y\in\Re^n} \left\{ f(y) + \frac{1}{2c}\|y - x\|^2 \right\}.$$

This iteration is done with respect to the second coordinate subvector x_2 of the vector $x = (x_1, x_2)$; it starts at the current vector (x_1^k, x_2^k), and yields a vector (x_1^k, x_2^{k+1}).

(b) An iteration of the ordinary proximal algorithm starting at the vector (x_1^k, x_2^{k+1}) obtained from the preceding coordinate descent iteration; this can be interpreted as a gradient iteration with stepsize equal to c for minimizing the same function F_c (see, e.g., [BeT89, p. 234]).

By contrast, the ordinary proximal iteration performs only step (b) above. Thus the partial iteration differs from the ordinary one only in that it executes an extra coordinate descent step prior to each ordinary proximal iteration. Note here that F_c

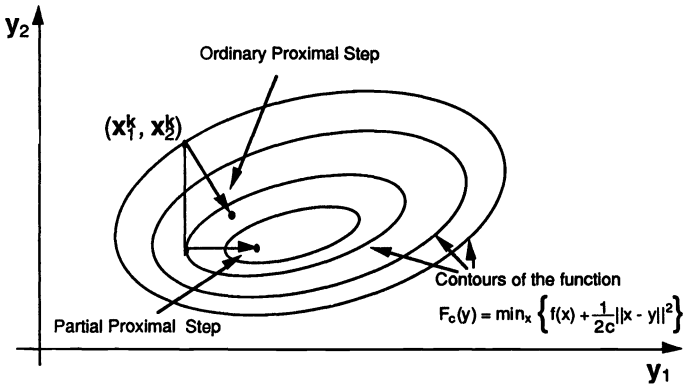


FIG. 1. *An illustration of the ordinary and the partial proximal iterations starting at the vector* $x^k = (x_1^k, x_2^k)$. *Both iterations involve a gradient step on the function* F_c *with stepsize* c. *However, the partial proximal iteration precedes the gradient step with a coordinate descent step along the subvector* x_2.

is continuously differentiable and has the same minimizers and minimum value as f; see e.g., [BeT89, §3.4.3]. Thus steps (a) and (b) above are both aimed at approaching an optimal solution.

A consequence of the preceding observation is that the value of the function F_c provides a uniform criterion of merit, which is improved by all partial proximal iterations, independently of the partition (x_1, x_2). We use this fact in §3 to provide a short convergence proof for a parallel algorithm that involves execution of different partial proximal iterations by different processors. In §4, we derive the rate of convergence of this algorithm. In §5, we show that partial proximal minimization algorithms are intimately related to augmented Lagrangian algorithms with partial elimination of constraints. When specialized within the augmented Lagrangian context, the parallel algorithm of §3 becomes similar to the parallel constraint distribution algorithms of Ferris and Mangasarian [FeM91], [Fer91a]; however, our convergence proofs are less complicated than those in [FeM91] and [Fer91a], and require less restrictive assumptions. In particular, the convergence analysis of Ferris and Mangasarian [FeM91], [Fer91a] assumes that the cost function is positive definite quadratic and the constraints are linear, while we assume general convex cost and constraint functions. Most of our analysis carries through to partial proximal minimization algorithms with nonquadratic proximal terms [Ber82], [CeZ92], [ChT93], [Eck93], [KoB76], [GoT79], [Luq84], [TsB93]. We thus take these more general methods as our starting point and specialize our results to the case of quadratic proximal terms whenever the results for this case are stronger. On the other hand, our analysis assumes that the proximal term contains the origin in its interior, and thus does not apply to methods using logarithmic/entropy proximal terms [Ber82], [CeZ92], [ChT93], [Eck93], and the corresponding augmented Lagrangian methods that use the exponential penalty function [KoB72], [Ber82], [TsB93].

2. A uniform descent property. In this section we introduce the notion of partial proximal minimization and analyze its descent properties. These descent properties will be used later to establish the convergence of algorithms based on successive applications of partial proximal minimization.

We first define partial proximal minimization in the general context of non-

quadratic proximal terms. Consider the class of strictly convex, continuously differentiable functions $\phi : \Re \to \Re$ such that

$$\phi(0) = 0, \quad \nabla\phi(0) = 0, \quad \lim_{t\to-\infty} \nabla\phi(t) = -\infty, \quad \lim_{t\to\infty} \nabla\phi(t) = \infty.$$

This class was introduced in [KoB76] within the dual context of nonquadratic augmented Lagrangian methods, together with the generalization of the proximal minimization algorithm obtained by replacing "$|\cdot|^2/2$" with "$\phi(\cdot)$." For an extensive discussion of the subject, see [Ber82, Chap. 5]. A prominent example in the class is the power function: $\phi(t) = (1/\gamma)|t|^\gamma$ with $\gamma > 1$. For $\gamma = 2$ we obtain the quadratic function used earlier.

For each $c > 0$, let F_c be the real-valued convex function on $\Re^n$ defined by

$$F_c(x) = \min_{y\in\Re^n} \left\{ f(y) + \frac{1}{c}\Phi(y - x) \right\}, \tag{3}$$

where $\Phi : \Re^n \mapsto \Re$ is the function

$$\Phi(t_i, \ldots, t_n) = \sum_{i=1}^{n} \phi(t_i).$$

We remark that we can have different ϕ's for different coordinate indices but, for simplicity, we do not consider this more general case. We also note that the gradient mapping $\nabla\Phi : \Re^n \mapsto \Re^n$ has an inverse $\nabla\Phi^{-1} : \Re^n \mapsto \Re^n$ because of the nature of the defining properties of the function Φ. We have the following proposition that formalizes the interpretation of the partial proximal iteration as a block coordinate descent step followed by an ordinary proximal step (compare with Fig. 1).

PROPOSITION 1. *Let $c > 0$ and a subset I of the index set $\{1, \ldots, n\}$ be given. For any $x \in \Re^n$, consider a vector x' satisfying*

$$x' \in \arg\min_{y\in\Re^n} \left\{ f(y) + \frac{1}{c}\sum_{i\in I} \phi(y_i - x_i) \right\}, \tag{4}$$

and let x'' be the vector with components

$$x_i'' = \begin{cases} x_i & \forall\, i \in I, \\ x_i' & \forall\, i \notin I. \end{cases} \tag{5}$$

Then

$$x' = x'' + \nabla\Phi^{-1}\left(-c\nabla F_c(x'')\right) = \arg\min_{y\in\Re^n} \left\{ f(y) + \frac{1}{c}\Phi(y - x'') \right\}, \tag{6}$$

$$x'' \in \arg\min_{\{y \mid y_i = x_i,\ i\in I\}} F_c(y), \tag{7}$$

where $F_c : \Re^n \to \Re$ is the convex function defined by (3). *Conversely if x' and x'' satisfy* (6) *and* (7), *then they also satisfy* (4) *and* (5).

Proof. Fix any $x \in \Re^n$. Let x' be a vector satisfying (4) and let x'' be given by (5). We will show that (6) and (7) hold. Indeed, from the definition of x'', the vector

x' minimizes not only $f(y) + \frac{1}{c}\sum_{i\in I}\phi(y_i - x_i)$, but also $\frac{1}{c}\sum_{i\notin I}\phi(y_i - x_i'')$, implying that x' minimizes the sum, which is $f(y)+\frac{1}{c}\Phi(y-x'')$. This proves the second equality in (6). The first equality in (6) is due to the invertibility of $\nabla\Phi$ (since $\Phi(\cdot)$ is strictly convex) and the following consequence of Proposition 5.5(c) in [Ber82]:

$$\nabla\Phi(x' - x'') = -c\nabla F_c(x'').$$

(Actually, Proposition 5.5(c) of Bertsekas [Ber82] addresses the dual context of nonquadratic augmented Lagrangian methods, and thus considers (dual) functions f with $f(x) = \infty$ for all x outside the nonnegative orthant. The proof given in [Ber82], however, applies verbatim to the more general convex function f considered here.)

To prove (7), note that for all vectors $z \in \Re^n$ with $z_i = x_i$ for all $i \in I$, we have

$$\begin{aligned}
F_c(z) &= \min_{y\in\Re^n}\left\{f(y) + \frac{1}{c}\sum_{i\in I}\phi(y_i - x_i) + \frac{1}{c}\sum_{i\notin I}\phi(y_i - z_i)\right\}\\
&\geq \min_{y\in\Re^n}\left\{f(y) + \frac{1}{c}\sum_{i\in I}\phi(y_i - x_i)\right\}\\
&= f(x') + \frac{1}{c}\sum_{i\in I}\phi(x_i' - x_i)\\
&= f(x') + \frac{1}{c}\sum_{i\in I}\phi(x_i' - x_i) + \frac{1}{c}\sum_{i\notin I}\phi(x_i' - x_i')\\
&\geq F_c(x''),
\end{aligned}$$

where the last inequality follows from the definitions of x'' and F_c. This proves (7).

Conversely, suppose that x' and x'' satisfy (6) and (7). We will show that (4) and (5) hold. Indeed, (7) implies that $x_i = x_i''$ for all $i \in I$, and that $\partial F_c(x'')/\partial x_i = 0$ for all $i \notin I$, so from (6) we have $x_i' = x_i''$ for all $i \notin I$. Thus (5) holds. To show (4) we argue by contradiction. Suppose that for some $z \in \Re^n$ we have

$$f(z) + \frac{1}{c}\sum_{i\in I}\phi(z_i - x_i) < f(x') + \frac{1}{c}\sum_{i\in I}\phi(x_i' - x_i).$$

Then the directional derivative of the function $y \mapsto f(y)+\frac{1}{c}\sum_{i\in I}\phi(y_i-x_i)$ at x' along the direction $z - x'$ is negative. This directional derivative is equal to the directional derivative of the function $y \mapsto f(y)+\frac{1}{c}\sum_{i\in I}\phi(y_i-x_i)+\frac{1}{c}\sum_{i\notin I}\phi(y_i-x_i')$ at x' along the direction $z-x'$. The latter directional derivative, however, is nonnegative in view of (5) and (6), arriving at a contradiction. This proves (4). □

By using Proposition 1 and the special structure of F_c, we obtain the following key descent property for F_c under the partial proximal iteration. This property will be used in the subsequent convergence analysis.

LEMMA 1. *Let $c > 0$ and a subset I of $\{1,\ldots,n\}$ be given. Let $F_c : \Re^n \to \Re$ be the continuously differentiable convex function given by* (3).

(a) *For any $x \in \Re^n$, the vector x' given by* (4) *satisfies*

$$F_c(x') - F_c(x'') \leq -\frac{1}{c}\Phi(x' - x''),$$

where x'' is the vector given by (5).

(b) *Let* $\phi(\cdot) = \frac{1}{2}|\cdot|^2$ *(so that* $\Phi(\cdot)$ *is the quadratic function* $\frac{1}{2}\|\cdot\|^2$*). Then, for any* $x \in \Re^n$*, the vector* x' *given by* (4) *satisfies*

$$F_c(x') - F_c(x) \leq -\frac{c}{4}||\nabla F_c(x)||^2.$$

Proof. (a). Fix any $x \in \Re^n$ and let x' and x'' be given by (4) and (5), respectively. By Proposition 1, x' and x'' satisfy (6), and so the definition of F_c yields

$$F_c(x'') = f(x') + \frac{1}{c}\Phi(x' - x'').$$

It is also easily seen from the definition of F_c that $F_c(x') \leq f(x')$ which, together with the above equality, implies the result.

(b) We first establish some basic inequalities satisfied by the function F_c and its gradient. For any $x, y \in \Re^n$, let

$$\tilde{x} = \arg\min_{z \in \Re^n} \left\{ f(z) + \frac{1}{2c}||z - x||^2 \right\}, \qquad \tilde{y} = \arg\min_{z \in \Re^n} \left\{ f(z) + \frac{1}{2c}||z - y||^2 \right\}.$$

Then,

$$\frac{1}{c}(x - \tilde{x}) \in \partial f(\tilde{x})$$

and, moreover, by (6) and $\Phi(\cdot) = \frac{1}{2}\|\cdot\|^2$,

$$x - \tilde{x} = c\nabla F_c(x), \qquad y - \tilde{y} = c\nabla F_c(y).$$

Combining these three relations and using the convexity of f, we obtain

$$\begin{aligned} & F_c(y) - F_c(x) - \langle \nabla F_c(x), y - x \rangle \\ & \quad = f(\tilde{y}) + \frac{1}{2c}||\tilde{y} - y||^2 - f(\tilde{x}) - \frac{1}{2c}||\tilde{x} - x||^2 - \frac{1}{c}\langle x - \tilde{x}, y - x \rangle \\ & \quad = f(\tilde{y}) - f(\tilde{x}) - \frac{1}{c}\langle x - \tilde{x}, \tilde{y} - \tilde{x} \rangle + \frac{1}{2c}||\tilde{y} - y - (\tilde{x} - x)||^2 \\ & \quad \geq \frac{1}{2c}||\tilde{y} - y - (\tilde{x} - x)||^2 \\ & \quad = \frac{c}{2}||\nabla F_c(y) - \nabla F_c(x)||^2 \qquad \forall\, x, y \in \Re^n. \end{aligned} \tag{8}$$

Let us now fix $x \in \Re^n$ and let x' and x'' be given by (4) and (5), respectively. By Proposition 1, x' and x'' satisfy (6), so the assumption $\Phi(\cdot) = \frac{1}{2}\|\cdot\|^2$ implies $x' = x'' - c\nabla F_c(x'')$. Then part (a) yields

$$F_c(x') - F_c(x'') \leq -\frac{1}{2c}||x' - x''||^2 = -\frac{c}{2}||\nabla F_c(x'')||^2.$$

Relation (7) implies $\langle \nabla F_c(x''), x'' - x \rangle = 0$ so, by invoking (8), we also obtain

$$\frac{c}{2}||\nabla F_c(x) - \nabla F_c(x'')||^2 \leq F_c(x) - F_c(x'').$$

Adding the above two relations and rearranging terms yields

$$\frac{c}{2}\left(||\nabla F_c(x) - \nabla F_c(x'')||^2 + ||\nabla F_c(x'')||^2\right) \leq F_c(x) - F_c(x'),$$

so, by using the following easily verifiable inequality

$$||\nabla F_c(x)||^2 \leq 2\big(||\nabla F_c(x) - \nabla F_c(x'')||^2 + ||\nabla F_c(x'')||^2\big),$$

the result follows. □

The following proposition provides some additional inequalities that compare the iterates of the ordinary and the partial proximal algorithms and that are useful for the convergence analysis of the latter (see the proof of Proposition 2).

LEMMA 2. *Let $c > 0$ and a subset I of $\{1,\dots,n\}$ be given. For any $x \in \Re^n$, the vector x' given by* (4) *satisfies*

$$f(x') \leq F_c(x) \leq f(x),$$

where $F_c : \Re^n \to \Re$ is the convex function defined by (3).

Proof. We have

$$\begin{aligned} f(x') &\leq f(x') + \frac{1}{c}\sum_{i\in I}\phi(x'_i - x_i) \\ &= \min_{y\in\Re^n}\left\{f(y) + \frac{1}{c}\sum_{i\in I}\phi(y_i - x_i)\right\} \\ &\leq \min_{y\in\Re^n}\left\{f(y) + \frac{1}{c}\Phi(y - x)\right\} \\ &\leq f(x). \end{aligned}$$

Since the expression in the right-hand side of the second inequality is equal to $F_c(x)$, the result follows. □

Note an important consequence of Lemma 2: If f is bounded below and if $\{x^k\}$ is a sequence generated by the partial proximal minimization iteration (2), then both $\{f(x^k)\}$ and $\{F_c(x^k)\}$ converge monotonically to the same value regardless of the particular partition used. It is possible to change the partition from one iteration to the next if this can improve convergence. Furthermore, if Φ is a quadratic function, we have from Lemma 1(b) that $\nabla F_c(x^k) \to 0$, so that all cluster points of $\{x^k\}$ minimize F_c and hence also f. This result is extended in the next section when we consider parallel versions of the partial proximal minimization algorithm.

3. Parallel algorithms. We now consider the following extension of the partial proximal minimization algorithm. At the start of the kth iteration we have the current iterate x^k. We construct all distinct partitions of x^k into two subvectors, and we execute the partial proximal iteration corresponding to each partition and to a chosen scalar c^k. The next iterate x^{k+1} is an arbitrary convex combination of the different vectors thus obtained. We describe below this algorithm, which we call the *parallel partial proximal minimization algorithm* (parallel PPM algorithm).

Let $\mathcal{C}$ denote the set of all subsets of $\{1,\dots,n\}$. Beginning with an arbitrary $x^0 \in \Re^n$, we generate a sequence $\{x^k\}$ as follows: Given x^k, we choose a scalar $c^k > 0$ and, for every subset $I \in \mathcal{C}$, let $\tilde{x}_I^k$ be given by

$$\tilde{x}_I^k \in \arg\min_{y\in\Re^n}\left\{f(y) + \frac{1}{c^k}\sum_{i\in I}\phi(y_i - x_i^k)\right\}; \tag{9}$$

we define x^{k+1} to be an arbitrary convex combination of the vectors $\tilde{x}_I^k$; that is,

$$x^{k+1} = \sum_{I \in \mathcal{C}} \alpha_I^k \tilde{x}_I^k, \tag{10}$$

where α_I^k, $I \in \mathcal{C}$, are any scalars satisfying

$$\sum_{I \in \mathcal{C}} \alpha_I^k = 1, \qquad \alpha_I^k \geq 0 \quad \forall\, I \in \mathcal{C}. \tag{11}$$

(Note that an α_I^k may be zero, so we need only to compute the vectors $\tilde{x}_I^k$ with $\alpha_I^k > 0$.)

For the parallel PPM algorithm to be well defined, we assume that the minimum in each partial proximal iteration (cf. (9)) is attained. Note that the multiple partial proximal iterations involved in (10)–(11) can be executed in parallel by multiple processors. The next proposition shows the validity of the algorithm by combining the inequalities of Lemmas 1 and 2, and by using a modification of the convergence arguments for the ordinary proximal minimization algorithm (see [BeT89, p. 240]).

PROPOSITION 2. *Let $\{x^k\}$ be a sequence generated by the parallel* PPM *algorithm with $\{c^k\}$ monotonically nondecreasing.*

(a) *If the set of minimizers of f is nonempty and compact, then $\{x^k\}$ is bounded, each of its cluster points is a minimizer of f, and $\lim_{k\to\infty} f(x^k) = \min_x f(x)$.*

(b) *Assume that f is bounded below and let $\phi(\cdot) = \frac{1}{2}|\cdot|^2$ (so $\Phi(\cdot)$ is the quadratic function $\frac{1}{2}\|\cdot\|^2$). Both $\{f(x^k)\}$ and $\{F_{c^k}(x^k)\}$ are monotonically nonincreasing and $\lim_{k\to\infty} \nabla F_{c^k}(x^k) = 0$, where $F_c : \Re^n \to \Re$ is the function given by* (3).

Proof. (a) For each k and $I \in \mathcal{C}$, we have by applying Lemma 2 with $x = x^k$, $c = c^k$, and $x' = \tilde{x}_I^k$ that

$$f(\tilde{x}_I^k) \leq F_{c^k}(x^k) \leq f(x^k).$$

This together with (10)–(11) and the convexity of f yields

$$f(x^{k+1}) \leq \sum_{I \in \mathcal{C}} \alpha_I^k f(\tilde{x}_I^k) \leq f(x^k).$$

Thus, $\{f(x^k)\}$ is monotonically nonincreasing.

For each k and $I \in \mathcal{C}$, since $\tilde{x}_I^k$ is given by (9), we have from Proposition 1 that

$$\tilde{x}_I^k = \arg\min_{y \in \Re^n} \left\{ f(y) + \frac{1}{c^k} \Phi(y - \hat{x}_I^k) \right\}, \tag{12}$$

where $\hat{x}_I^k$ is the vector in $\Re^n$ whose ith component is the ith component of x^k if $i \in I$ and, otherwise, is the ith component of $\tilde{x}_I^k$. We also have from (7) that, for each k and $I \in \mathcal{C}$,

$$F_{c^k}(\hat{x}_I^k) \leq F_{c^k}(x^k),$$

and from Lemma 1(a) that

$$F_{c^k}(\tilde{x}_I^k) \leq F_{c^k}(\hat{x}_I^k) - \frac{1}{c^k} \Phi(\tilde{x}_I^k - \hat{x}_I^k).$$

Combining the above two relations and using (10)–(11) and the convexity of F_{c^k} yields

$$F_{c^k}(x^{k+1}) \leq F_{c^k}(x^k) - \frac{1}{c^k} \sum_{I \in \mathcal{C}} \alpha_I^k \Phi(\tilde{x}_I^k - \hat{x}_I^k).$$

Since $F_{c^{k+1}}(x^{k+1}) \leq F_{c^k}(x^{k+1})$ (cf. $c^{k+1} \geq c^k$ and (3)), this shows that $\{F_{c^k}(x^k)\}$ is monotonically nonincreasing and that

$$\lim_{k\to\infty} \frac{1}{c^k} \sum_{I\in\mathcal{C}} \alpha_I^k \Phi(\tilde{x}_I^k - \hat{x}_I^k) = 0. \tag{13}$$

Also, the compactness of the set of minimizers of f implies that all the level sets of f are compact [Roc70, Cor. 8.7.1] and that, for $k \geq 1$ and $I \in \mathcal{C}$, x^k and $\tilde{x}_I^k$ are contained in the level set $\{x \mid f(x) \leq f(x^1)\}$ (cf. Lemma 2). It follows that the sequences $\{x^k\}$ and $\{\tilde{x}_I^k\}$ are bounded. Since each component of $\hat{x}_I^k$ is either a component of x^k or of $\tilde{x}_I^k$, it follows that the sequence $\{\hat{x}_I^k\}$ is also bounded.

Fix any minimizer x^* of f. Using (12) and the convexity of f, we have for each $\xi \in (0,1)$ and $I \in \mathcal{C}$ that

$$\begin{aligned} f(\tilde{x}_I^k) + \frac{1}{c^k}\Phi(\tilde{x}_I^k - \hat{x}_I^k) &\leq f\big(\xi x^* + (1-\xi)\tilde{x}_I^k\big) + \frac{1}{c^k}\Phi\big(\tilde{x}_I^k - \hat{x}_I^k + \xi(x^* - \tilde{x}_I^k)\big) \\ &\leq \xi f(x^*) + (1-\xi) f(\tilde{x}_I^k) + \frac{1}{c^k}\Phi\big(\tilde{x}_I^k - \hat{x}_I^k + \xi(x^* - \tilde{x}_I^k)\big). \end{aligned}$$

Rearranging terms and dividing both sides by ξ gives

$$f(\tilde{x}_I^k) \leq f(x^*) + \frac{1}{\xi c^k}\Phi\big(\tilde{x}_I^k - \hat{x}_I^k + \xi(x^* - \tilde{x}_I^k)\big),$$

which together with (10)–(11) and the convexity of f yields

$$\begin{aligned} f(x^{k+1}) &\leq f(x^*) + \frac{1}{\xi c^k} \sum_{I\in\mathcal{C}} \alpha_I^k \Phi\big(\tilde{x}_I^k - \hat{x}_I^k + \xi(x^* - \tilde{x}_I^k)\big) \\ &\leq f(x^*) + \frac{1}{\xi c^k} \sum_{I\in\mathcal{C}} \alpha_I^k \left[\Phi(\tilde{x}_I^k - \hat{x}_I^k) + \xi\langle \nabla\Phi\big(\tilde{x}_I^k - \hat{x}_I^k + \xi(x^* - \tilde{x}_I^k)\big), x^* - \tilde{x}_I^k\rangle\right], \end{aligned}$$

where the second inequality follows from the convexity of Φ. By taking the limit supremum as $k \to 0$ and using (13), we obtain

$$\limsup_{k\to\infty} f(x^{k+1}) \leq f(x^*) + \limsup_{k\to\infty} \frac{1}{c^k} \sum_{I\in\mathcal{C}} \alpha_I^k \langle \nabla\Phi\big(\tilde{x}_I^k - \hat{x}_I^k + \xi(x^* - \tilde{x}_I^k)\big), x^* - \tilde{x}_I^k\rangle.$$

Finally, we take the limit of both sides as $\xi \to 0$. Since the sequences $\{\tilde{x}_I^k\}$, $\{\hat{x}_I^k\}$, and $\{1/c^k\}$ are all bounded, we can pass this limit through the limit supremum on the right-hand side to obtain

$$\limsup_{k\to\infty} f(x^{k+1}) \leq f(x^*) + \limsup_{k\to\infty} \frac{1}{c^k} \sum_{I\in\mathcal{C}} \alpha_I^k \langle \nabla\Phi\big(\tilde{x}_I^k - \hat{x}_I^k\big), x^* - \tilde{x}_I^k\rangle. \tag{14}$$

Assume first that $c^k \to \infty$. Since both sequences $\{\tilde{x}_I^k\}$ and $\{\hat{x}_I^k\}$ are bounded, it follows from (14) that $\limsup_{k\to\infty} f(x^{k+1}) \leq f(x^*)$. Since x^* minimizes f, $\{f(x^k)\}$ must converge to $f(x^*)$.

Assume now that $c^k \to \bar{c} < \infty$. Let $\mathcal{K}$ be an infinite subsequence of the set of positive integers such that for every $I \in \mathcal{C}$, $\{\alpha_I^k\}_{k\in\mathcal{K}}$ converges (to either a positive

number or zero) with $\lim_{k\to\infty,\ k\in\mathcal{K}} \alpha_I^k > 0$ for at least one $I \in \mathcal{C}$; since $\mathcal{C}$ is a finite set, such a subsequence exists. From (13) we see that for all $I \in \mathcal{C}$ such that $\lim_{k\to\infty,\ k\in\mathcal{K}} \alpha_I^k > 0$, we have

$$\lim_{k\to\infty,\ k\in\mathcal{K}} \Phi(\tilde{x}_I^k - \hat{x}_I^k) = 0$$

and hence also $\lim_{k\to\infty,\ k\in\mathcal{K}} \{\tilde{x}_I^k - \hat{x}_I^k\} = 0$, implying that

$$\lim_{k\to\infty,\ k\in\mathcal{K}} \nabla\Phi(\tilde{x}_I^k - \hat{x}_I^k) = 0.$$

Using the above equation in (14), we obtain $\limsup_{k\to\infty,\ k\in\mathcal{K}} f(x^{k+1}) \le f(x^*)$. Since x^* minimizes f, $\{f(x^k)\}_{k\in\mathcal{K}}$ must converge to $f(x^*)$. Since $\{f(x^k)\}$ is monotonically nonincreasing, it too must converge to $f(x^*)$.

(b) Fix any k. For each $I \in \mathcal{C}$, we have upon applying Lemma 1(b) with $x = x^k$, $c = c^k$, and x' equal to the vector $\tilde{x}_I^k$ given by (9) that

$$F_{c^k}(\tilde{x}_I^k) - F_{c^k}(x^k) \le -\frac{c^k}{4}||\nabla F_{c^k}(x^k)||^2.$$

Combining this with (10)–(11) and the convexity of F_{c^k}, we obtain

$$\begin{aligned} F_{c^k}(x^{k+1}) - F_{c^k}(x^k) &\le \sum_{I\in\mathcal{C}} \alpha_I^k \big(F_{c^k}(\tilde{x}_I^k) - F_{c^k}(x^k)\big) \\ &\le -\frac{c^k}{4} \sum_{I\in\mathcal{C}} \alpha_I^k ||\nabla F_{c^k}(x^k)||^2 \\ &= -\frac{c^k}{4}||\nabla F_{c^k}(x^k)||^2. \end{aligned}$$

Since $F_{c^{k+1}}(x^{k+1}) \le F_{c^k}(x^{k+1})$ (cf. $c^{k+1} \ge c^k$ and (3)), and F_{c^k} is bounded below by $\inf_{x\in\Re^n} f(x)$ for every k, the preceding relation implies that $\{F_{c^k}(x^k)\}$ is monotonically nonincreasing and that

$$\lim_{k\to\infty} c^k||\nabla F_{c^k}(x^k)||^2 = 0.$$

Since $\{c^k\}$ is bounded below by $c^0 > 0$, this proves that $\lim_{k\to\infty} \nabla F_{c^k}(x^k) = 0$.

For each $I \in \mathcal{C}$, we have upon applying Lemma 2 with $x = x^k$, $c = c^k$, and (cf. (9)) $x' = \tilde{x}_I^k$ that

$$f(\tilde{x}_I^k) \le F_{c^k}(x^k) \le f(x^k).$$

This together with (10)–(11) and the convexity of f yields

$$f(x^{k+1}) \le \sum_{I\in\mathcal{C}} \alpha_I^k f(\tilde{x}_I^k) \le F_{c^k}(x^k) \le f(x^k).$$

Thus, $\{f(x^k)\}$ is monotonically nonincreasing. □

It can be seen from the above proof that the monotonicity property of $\{c^k\}$ is not crucial for Proposition 2 to hold. Instead, it suffices that $\{F_{c^k}(x^k)\}$ is maintained nonincreasing and $\{c^k\}$ is bounded away from zero.

The parallel PPM algorithm can be generalized and implemented in a flexible and asynchronous manner. One can envision multiple processors executing asynchronously different partial proximal iterations starting from the vector that is currently best in

terms of some uniform merit criterion, such as the value of f or F_c. The results from several processors can be combined via a convex combination, but the convex combination should be accepted only if the uniform merit criterion is improved.

It should be noted that the above convergence results for the parallel PPM algorithm are considerably weaker than those available for the proximal minimization algorithm (see [BeT89, §3.4.3], [ChT93], [EcB92], [Fer91b], [GoT79], [Gul91], [Luq86], [Mar70], [Roc76]). One difficulty is that the extra coordinate descent step of (7) destroys certain monotonicity properties of the iterates that are essential to proving the strong convergence properties of the proximal minimization algorithm.

Finally, we note that the choice of the coefficients α_I^k, $I \in \mathcal{C}$ can have a significant effect on the convergence rate of the parallel PPM algorithm. To illustrate, consider applying the parallel PPM algorithm with $\phi(\cdot) = \frac{1}{2}|\cdot|^2$ to minimize the two-dimensional convex differentiable function

$$f(x_1, x_2) = \frac{1}{2}\left(\max\{0, x_1\}^2 + \max\{0, x_2\}^2\right).$$

Suppose furthermore that $x_1^0 > 0$ and $x_2^0 > 0$ and that $c^k = c > 0$ for all k. If we set $\alpha_{\{1\}}^k = 1$ (so $\alpha_{\{2\}}^k = \alpha_{\{1,2\}}^k = 0$) for all k, then it is not difficult to see that one possible sequence is given by $x_1^k = x_1^0/(1+c)^k$ and $x_2^k = (-1)^k - 1$ for all k, so the cost converges at a linear rate, but the iterates themselves do not converge. On the other hand, if we set $\alpha_{\{1\}}^k$ to alternate between 1 and 0 with the corresponding values of $\alpha_{\{2\}}^k$ alternating between 0 and 1, while $\alpha_{\{1,2\}}^k = 0$, then it can be seen that a minimizer of f is obtained after only two iterations.

4. Rate of convergence. We now turn to the analysis of the convergence rate of the parallel PPM algorithm. To establish some terminology, consider a real sequence $\{s_k\}$ that converges to a real number s^*. We say that $\{s_k\}$ converges *finitely* if there exists $\overline{k}$ such that $s_k = s^*$ for all $k \geq \overline{k}$; *superlinearly with order p*, where $p > 1$, if

$$\limsup_{k\to\infty} \frac{|s_{k+1} - s^*|}{|s_k - s^*|^p} < \infty;$$

superlinearly if

$$\lim_{k\to\infty} \frac{|s_{k+1} - s^*|}{|s_k - s^*|} = 0;$$

and *linearly* if there exist $\beta \in (0,1)$ and $\overline{k}$ such that

$$\frac{|s_{k+1} - s^*|}{|s_k - s^*|} \leq \beta \quad \forall\, k \geq \overline{k}.$$

Following [OrR70], we say that $\{s_k\}$ converges *R-linearly* if there exist scalars $q > 0$ and $\beta \in [0,1)$ such that $|s_k - s^*| \leq q\beta^k$ for all k. Note that $\{s_k\}$ converges R-linearly if $|s_k - s^*| \leq t_k$ for all k where $\{t_k\}$ is some sequence converging linearly to zero.

The convergence rate of the ordinary proximal minimization algorithm depends on the growth properties of the minimized function f as well as on the growth properties of the proximal term used. The following key assumption was first introduced in [KoB76] (see also [Ber82, p. 342]) and was used to analyze the convergence rate of the proximal minimization algorithm for quadratic as well as certain types of nonquadratic proximal terms.

Assumption A. The set of minimizers of f, denoted X^*, is nonempty and compact. Furthermore, there exist scalars $\alpha \geq 1$, $\beta > 0$, and $\delta > 0$ such that

$$\beta\big(\rho(x; X^*)\big)^{\alpha} \leq f(x) - \min_{y \in \Re^n} f(y) \qquad \forall\, x \text{ with } \rho(x; X^*) \leq \delta, \tag{15}$$

where $\rho(x; X^*)$ is the distance from x to X^* given by

$$\rho(x; X^*) = \min_{x^* \in X^*} ||x - x^*||. \tag{16}$$

The ordinary proximal minimization algorithm has finite, superlinear, or linear convergence rate depending on whether $\alpha = 1$, $1 < \alpha < 2$, or $\alpha = 2$, respectively; see [KoB76] and also [Ber75] (which deals with the case $\alpha = 1$), and [Ber82, Chap. 5] (that provides a comprehensive analysis). The convergence rate is also superlinear if $\alpha = 2$ and $c^k \to \infty$. If $\alpha > 2$, the convergence rate is slower than linear, that is, some of the generated sequences do not converge linearly. In the case where the proximal term has a growth rate $\gamma > 1$ other than quadratic ($\gamma \neq 2$), the convergence rate is influenced by γ (it is superlinear if $1 < \alpha < \gamma$ even in the case where $\alpha \geq 2$).

The following proposition provides corresponding, although slightly weaker, results for the parallel PPM algorithm.

PROPOSITION 3. *Let Assumption* A *hold, let* $f^* = \min_{y \in \Re^n} f(y)$, *and let* $\{x^k\}$ *be a sequence generated by the parallel* PPM *algorithm with* $\{c^k\}$ *monotonically nondecreasing.*

(a) *If* $\alpha = 1$ *and there exists a scalar* $\overline{\alpha} > 0$ *such that*

$$\alpha_I^k \geq \overline{\alpha} \quad \forall\, k \quad \text{and} \quad I \in \mathcal{C} \text{ such that } \alpha_I^k > 0, \tag{17}$$

then $\{f(x^k)\}$ *converges to* f^* *finitely.*

(b) *Assume that for some scalars* $M > 0$ *and* $\gamma \geq \alpha$ *we have*

$$\Phi(x) \leq M||x||^{\gamma} \quad \forall\, x \text{ with } ||x|| \leq \delta. \tag{18}$$

If $1 < \alpha < \gamma$, *then* $\{f(x^k)\}$ *converges to* f^* *superlinearly with order* γ/α. *Also, if* $1 < \alpha = \gamma$ *and* $c^k \to \infty$, *then* $\{f(x^k)\}$ *converges to* f^* *superlinearly.*

(c) *Let* $\phi(\cdot) = \frac{1}{2}|\cdot|^2$ (*so* $\Phi(\cdot)$ *is the quadratic function* $\frac{1}{2}\|\cdot\|^2$). *If* $\alpha = 2$ *and* $c^k \to \overline{c} < \infty$, *then* $\{f(x^k)\}$ *converges to* f^* *R-linearly.*

Proof. By Proposition 2(a), we have $f(x^k) \to f^*$ and, by using also Lemma 2, we have $f(\tilde{x}_I^k) \to f^*$ for all $I \in \mathcal{C}$. Since X^* is compact, it follows that for all k sufficiently large, we have

$$\rho(x^k; X^*) \leq \delta, \qquad \rho(\tilde{x}_I^k; X^*) \leq \delta \quad \forall\, I \in \mathcal{C}.$$

Without loss of generality we assume that the above relation holds for all k.

(a) For each k and $I \in \mathcal{C}$, we have from Proposition 1 that $\tilde{x}_I^k$ minimizes $f(y) + \frac{1}{c^k}\Phi(y - \hat{x}_I^k)$ over y, where $\hat{x}_I^k$ is the vector in $\Re^n$ whose ith component is the ith component of x^k if $i \in I$ and, otherwise, is the ith component of $\tilde{x}_I^k$. Thus

$$g_I^k + \frac{1}{c^k}\nabla\Phi(\tilde{x}_I^k - \hat{x}_I^k) = 0, \tag{19}$$

where g_I^k is a subgradient of f at $\tilde{x}_I^k$. Let us denote by $\overline{x}_I^k$ the vector of X^*, which is at minimum distance from $\tilde{x}_I^k$, that is,

$$\rho(\tilde{x}_I^k; X^*) = ||\overline{x}_I^k - \tilde{x}_I^k|| \quad \text{and} \quad f(\overline{x}_I^k) = f^*.$$

Since $\rho(\tilde{x}_I^k; X^*) \leq \delta$, from (15) with $\alpha = 1$ and using the convexity of f, we have

$$\begin{aligned}\beta\rho(\tilde{x}_I^k; X^*) &\leq f(\tilde{x}_I^k) - f(\overline{x}_I^k)\\ &\leq \langle g_I^k, \tilde{x}_I^k - \overline{x}_I^k\rangle\\ &\leq ||g_I^k||\ ||\tilde{x}_I^k - \overline{x}_I^k||\\ &= ||g_I^k||\ \rho(\tilde{x}_I^k; X^*).\end{aligned}$$

Thus for all k and $I \in \mathcal{C}$ we have

$$0 \leq \big(||g_I^k|| - \beta\big)\rho(\tilde{x}_I^k; X^*). \tag{20}$$

If $c^k \to \infty$, then from (19) and the boundedness of $\{\tilde{x}_I^k - \hat{x}_I^k\}$, we have $g_I^k \to 0$ for all $I \in \mathcal{C}$, and (20) implies that for sufficiently large k, we have $\rho(\tilde{x}_I^k; X^*) = 0$ or equivalently $\tilde{x}_I^k \in X^*$, for all $I \in \mathcal{C}$. This implies that $x^{k+1} \in X^*$ for all sufficiently large k, so the algorithm terminates finitely.

If $c^k \to \bar{c} < \infty$, let $\mathcal{K}$ be an infinite subsequence of the set of positive integers such that for every $I \in \mathcal{C}$, either $\alpha_I^k > 0$ for all $k \in \mathcal{K}$ or $\alpha_I^k = 0$ for all $k \in \mathcal{K}$; since $\mathcal{C}$ is a finite set, such a subsequence exists. Let $\overline{\mathcal{C}}$ be the subset of index sets $I \in \mathcal{C}$ such that $\alpha_I^k > 0$ for all $k \in \mathcal{K}$. Using the assumption (17), we have

$$\alpha_I^k \geq \overline{\alpha} \qquad \forall k \in \mathcal{K} \quad \text{and} \quad I \in \overline{\mathcal{C}}, \qquad \alpha_I^k = 0 \quad \forall\, k \in \mathcal{K} \quad \text{and} \quad I \notin \overline{\mathcal{C}}.$$

Then from (13) we obtain

$$\lim_{k\to\infty,\ k\in\mathcal{K}} \sum_{I\in\overline{\mathcal{C}}} \Phi\big(\tilde{x}_I^k - \hat{x}_I^k\big) = 0,$$

implying

$$\lim_{k\to\infty,\ k\in\mathcal{K}} \Phi\big(\tilde{x}_I^k - \hat{x}_I^k\big) = 0 \quad \forall\, I \in \overline{\mathcal{C}}.$$

Therefore, we have

$$\lim_{k\to\infty,\ k\in\mathcal{K}} \nabla\Phi\big(\tilde{x}_I^k - \hat{x}_I^k\big) = 0 \quad \forall\, I \in \overline{\mathcal{C}},$$

or equivalently, in view of (19),

$$\lim_{k\to\infty,\ k\in\mathcal{K}} g_I^k = 0 \quad \forall\, I \in \overline{\mathcal{C}}.$$

It follows from (20) that for all $I \in \overline{\mathcal{C}}$ and all $k \in \mathcal{K}$ sufficiently large we have $\rho(\tilde{x}_I^k; X^*) = 0$. Using the definition $x^{k+1} = \sum_{i\in\mathcal{C}} \alpha_I^k \tilde{x}_I^k$ and the fact $\alpha_I^k = 0$ for all $I \notin \overline{\mathcal{C}}$ and $k \in \mathcal{K}$, we obtain $x^{k+1} \in X^*$ for all $k \in \mathcal{K}$ sufficiently large. Since the choice of the subsequence $\mathcal{K}$ was arbitrary and a finite number of such subsequences comprise all integers beyond some index, this shows that the algorithm terminates finitely.

(b) For each k, let us denote by $\overline{x}^k$ the vector of X^*, which is at minimum distance from x^k, that is,

$$\rho(x^k; X^*) = ||\overline{x}^k - x^k|| \quad \text{and} \quad f(\overline{x}^k) = f^*.$$

We have from (9) that for each $I \in \mathcal{C}$,

$$\begin{aligned} f(\tilde{x}_I^k) - f^* &\le f(\overline{x}^k) + \frac{1}{c^k}\sum_{i\in I}\phi(\overline{x}_i^k - x_i^k) - f^* \\ &\le \frac{1}{c^k}\Phi(\overline{x}^k - x^k) \\ &\le \frac{M}{c^k}||\overline{x}^k - x^k||^\gamma \\ &= \frac{M}{c^k}\rho(x^k; X^*)^\gamma \\ &\le \frac{M}{c^k}\left(\frac{f(x^k) - f^*}{\beta}\right)^{\gamma/\alpha}, \end{aligned}$$

where the third inequality follows from (18) and the last inequality follows from (15) and $\rho(x^k); X^*) \le \delta$. By using the convexity of f and the definition $x^{k+1} = \sum_{I\in\mathcal{C}} \alpha_I^k x_I^k$, we obtain for all k that

$$\frac{f(x^{k+1}) - f^*}{\left(f(x^k) - f^*\right)^{\gamma/\alpha}} \le \frac{M}{c^k \beta^{\gamma/\alpha}}. \tag{21}$$

This proves the result.

(c) Using the compactness of X^*, we have that there exists $\delta' > 0$ such that $\rho(x; X^*) \le \delta$ for all x such that $f(x) - f^* \le \delta'$. Given $x \in \Re^n$ such that $f(x) - f^* \le \delta'$, $c > 0$, and a subset $I \subset \{1, \ldots, n\}$, let x' and x'' be given by (4) and (5), respectively. By Proposition 1, x' satisfies (6) so

$$f(x') + \frac{1}{c}\langle x' - x'', y - x'\rangle \le f(y) \quad \forall\, y.$$

Let x^* be the element of X^* for which $||x' - x^*|| = \rho(x'; X^*)$. Setting $y = x^*$ into the above relation and rearranging terms, we obtain

$$f(x') - f^* \le \frac{1}{c}\langle x'' - x', x^* - x'\rangle \le \frac{1}{c}||x'' - x'||||x^* - x'||.$$

By Lemma 2 and the assumption $f(x) - f^* \le \delta'$, we have $f(x') - f^* \le \delta'$, so that $\rho(x'; X^*) \le \delta$ and (15) with $\alpha = 2$ yields

$$\beta||x^* - x'||^2 \le f(x') - f^* \le \frac{1}{c}||x'' - x'||||x^* - x'||$$

or, equivalently,

$$||x^* - x'|| \le \frac{1}{c\beta}||x'' - x'||.$$

Then, by (3) and $\Phi(\cdot) = \frac{1}{2}\|\cdot\|^2$,

$$F_c(x') - f^* \le f(x^*) + \frac{1}{2c}||x^* - x'||^2 - f^* \le \frac{1}{2c(c\beta)^2}||x'' - x'||^2.$$

On the other hand, we have from the proof of Lemma 1(b) that

$$F_c(x') \le F_c(x'') - \frac{1}{2c}||x' - x''||^2,$$

which together with the above relation yields

$$F_c(x') - f^* \leq \frac{1}{(c\beta)^2}\big(F_c(x'') - F_c(x')\big).$$

Rearranging terms and using the fact (cf. (7)) $F_c(x'') \leq F_c(x)$, we finally obtain

$$F_c(x') - f^* \leq \frac{1}{(c\beta)^2 + 1}\big(F_c(x) - f^*\big). \tag{22}$$

Consider now a sequence $\{x^k\}$ generated by the parallel PPM algorithm with $\{c^k\}$ monotonically nondecreasing. Since X^* is compact, by Proposition 2(a), we have $f(x^k) \to f^*$ so that $f(x^k) - f^* \leq \delta'$ for all k large enough. Fix any such k. By (9), we can apply (22) with $x = x^k$, $c = c^k$, and $x' = \tilde{x}_I^k$ for every $I \in \mathcal{C}$ and obtain

$$F_{c^k}(\tilde{x}_I^k) - f^* \leq \frac{1}{(c^k\beta)^2 + 1}\big(F_{c^k}(x^k) - f^*\big) \quad \forall\, I \in \mathcal{C}.$$

Then, using the definition (10)–(11) of x^{k+1} and the convexity of F_{c^k}, we obtain

$$F_{c^k}(x^{k+1}) - f^* \leq \frac{1}{(c^k\beta)^2 + 1}\big(F_{c^k}(x^k) - f^*\big) \quad \forall\, I \in \mathcal{C}.$$

Since $F_{c^{k+1}}(x^{k+1}) \leq F_{c^k}(x^{k+1})$ (cf. $c^{k+1} \geq c^k$ and (3)), this implies

$$F_{c^{k+1}}(x^{k+1}) - f^* \leq \frac{1}{(c^k\beta)^2 + 1}\big(F_{c^k}(x^k) - f^*\big).$$

We have that $\{c^k\}$ is bounded below by c^0, so it follows that $\{F_{c^k}(x^k)\}$ converges linearly. Since $f(x^{k+1}) \leq F_{c^k}(x^k)$ for all k, we obtain that $\{f(x^k)\}$ converges R-linearly and part (c) is proven. □

The preceding proof also shows that if $\alpha = \gamma$ and $c^k \to \bar{c} \in (\beta/M, \infty)$, then $\{f(x^k)\}$ converges linearly rather than R-linearly (see (21)).

The preceding analysis assumes that the set of minimizers of f is bounded. We show below that this assumption can be removed if the minimized function f is differentiable on its effective domain and has a growth property similar to that given by (15) with $\alpha = 2$. This result will be useful when we analyze dual applications of the partial proximal algorithm in §5, for which the set of minimizers of f is frequently unbounded (see Proposition 7).

Assumption B. The set of minimizers of f, denoted X^*, is nonempty and f has the special form:

$$f(x) = \begin{cases} g(x) & \text{if } x \in C, \\ \infty & \text{otherwise,} \end{cases} \tag{23}$$

where C is a nonempty closed convex set in $\Re^n$ and $g : \Re^n \mapsto \Re$ is a convex differentiable function. Furthermore, there exist scalars $\beta > 0$ and $\delta > 0$ such that

$$\beta\rho(x; X^*) \leq ||x - P_C[x - \nabla g(x)]|| \quad \forall\, x \in C \text{ with } ||x - P_C[x - \nabla g(x)]|| \leq \delta, \tag{24}$$

where $P_C[\cdot]$ denotes the orthogonal projection onto C and $\rho(x; X^*)$ is the distance from x to X^* defined by (16).

The growth condition (24) differs from the growth condition (15) (with $\alpha = 2$) mainly in that the cost difference $f(x) - f^*$ is approximated by the norm of a certain residual function squared. This difference is nonetheless significant for it turns out that the partial proximal algorithm drives the latter to zero even when X^* is unbounded (see the proof below). In general, verifying that condition (24) holds is not easy. However, it is known that this condition holds when g is strongly convex [Pan87] or when C is a polyhedral set and g is the composition of a strongly convex function with an affine mapping [LuT92]. (See [LuT93] and §5 for additional discussions of this condition.)

PROPOSITION 4. *Let Assumption* B *hold and let* $\{x^k\}$ *be a sequence generated by the parallel* PPM *algorithm* (9)–(11) *with* $\{c^k\}$ *monotonically nondecreasing, bounded above, and with* $\phi(\cdot) = \frac{1}{2}|\cdot|^2$. *Then* $\{f(x^k)\}$ *converges to* $\min_x f(x)$ *R-linearly.*

Proof. First, we show that, for any $c > 0$, the function F_c inherits from f a property similar to the growth condition (24). Fix any $x \in \Re^n$ and let

$$\tilde{x} = \arg\min_{y \in \Re^n} \left\{ f(y) + \frac{1}{2c}\|y - x\|^2 \right\}.$$

Then, by (6) and $\Phi(\cdot) = \frac{1}{2}\|\cdot\|^2$, we have $x - \tilde{x} = c\nabla F_c(x)$ and, by using (23), we also have that $\tilde{x}$ is a minimizer of the function $y \mapsto g(y) + \frac{1}{2c}\|y - x\|^2$ over C, so that

$$\tilde{x} = P_C[x - c\nabla g(\tilde{x})].$$

Thus,

$$||\tilde{x} - P_C[\tilde{x} - c\nabla g(\tilde{x})]|| = ||P_C[x - c\nabla g(\tilde{x})] - P_C[\tilde{x} - c\nabla g(\tilde{x})]|| \le ||x - \tilde{x}||,$$

where the second inequality follows from the nonexpansive property of the projection operator $P_C[\cdot]$. Since $||\tilde{x} - P_C[\tilde{x} - cd]|| \ge \hat{c}||\tilde{x} - P_C[\tilde{x} - d]||$ for any $d \in \Re^n$, where $\hat{c} = \min\{1, c\}$ (see Lemma 1 in [GaB84]), this implies

$$\hat{c}||\tilde{x} - P_C[\tilde{x} - \nabla g(\tilde{x})]|| \le ||x - \tilde{x}||.$$

Then, it readily follows from (24) that

$$\hat{c}\beta\rho(\tilde{x}; X^*) \le ||x - \tilde{x}|| \quad \text{whenever } ||x - \tilde{x}|| \le \hat{c}\delta.$$

Since $x - \tilde{x} = c\nabla F_c(x)$ and, by the triangle inequality, $\rho(x; X^*) \le ||x - \tilde{x}|| + \rho(\tilde{x}; X^*)$, this shows that

$$\beta\rho(x; X^*) \le (c\beta + c/\hat{c})||\nabla F_c(x)|| \quad \text{whenever } ||\nabla F_c(x)|| \le \delta c/\hat{c}.$$

Also, denoting $f^* = \min_y f(y)$, we have from (3) and $\Phi(\cdot) = \frac{1}{2}\|\cdot\|^2$ that

$$F_c(x) = f(\tilde{x}) + \frac{1}{2c}\|\tilde{x} - x\|^2 \le f^* + \frac{1}{2c}\|x^* - x\|^2 \quad \forall\, x^* \in X^*,$$

so that $F_c(x) \le f^* + \frac{1}{2c}\rho(x; X^*)^2$. Combining this with the previous relation yields

$$F_c(x) \le f^* + \frac{1}{2c}(c + c/(\hat{c}\beta)^2||\nabla F_c(x)||^2 \qquad \forall\, x \text{ with } ||\nabla F_c(x)|| \le \delta c/\hat{c}. \tag{25}$$

Consider now a sequence $\{x^k\}$ generated by the parallel PPM algorithm with $\{c^k\}$ monotonically nondecreasing and bounded above. By Proposition 2(b), we have $\nabla F_{c^k}(x^k) \to 0$, so that $||\nabla F_{c^k}(x^k)|| \leq \delta \hat{c}^k / c^k$ for all k sufficiently large, where $\hat{c}^k = \min\{1, c^k\}$. For any such k we have from (25) that

$$F_{c^k}(x^k) \leq f^* + \frac{1}{2c^k}(c^k + c^k/(\hat{c}^k\beta))^2 ||\nabla F_{c^k}(x^k)||^2.$$

In addition, by following the proof of Proposition 2(b), we see that

$$F_{c^{k+1}}(x^{k+1}) - F_{c^k}(x^k) \leq -\frac{c^k}{4}||\nabla F_{c^k}(x^k)||^2.$$

Combining the above two relations and rearranging terms yields

$$F_{c^{k+1}}(x^{k+1}) - f^* \leq \left(1 - \frac{(\hat{c}^k\beta)^2}{2(\hat{c}^k\beta + 1)^2}\right)(F_{c^k}(x^k) - f^*).$$

We have that $\{\hat{c}^k\}$ is bounded below by $\hat{c}^0$, so it follows that $\{F_{c^k}(x^k)\}$ converges linearly to f^*. Since $f(x^{k+1}) \leq F_{c^k}(x^k)$ for all k, we obtain that $\{f(x^k)\}$ converges R-linearly to f^*. □

5. Relation to multiplier methods. We assume throughout this section that $\phi(\cdot) = \frac{1}{2}|\cdot|^2$ (so that $\Phi(\cdot)$ is the quadratic function $\frac{1}{2}\|\cdot\|^2$), and we show that partial proximal iterations correspond to augmented Lagrangian iterations with partial elimination of constraints. This indicates a possible application area of the parallel PPM algorithm of §3 and establishes its relation to the constraint distribution method of Ferris and Mangasarian [FeM91]. In addition, by applying the convergence results of §4, we analyze the rate of convergence of these augmented Lagangian iterations under much weaker assumptions than those given in [FeM91]. For example, we establish linear rate of convergence for the cost of the iterates, assuming that the constraint functions are affine and the cost function is the sum of the indicator function for a polyhedral set and a strongly convex differentiable function with Lipschitz continuous gradient (see Proposition 7). In contrast, the linear rate of convergence result in [FeM91] in addition assumes that the cost function is quadratic. (On the other hand, the analysis in [FeM91] establishes the stronger result of linear rate of convergence of the iterates.)

Consider the following convex program

$$\text{minimize } h_0(z) \tag{26a}$$

$$\text{subject to } h_1(z) \leq 0, \ldots, h_n(z) \leq 0, \tag{26b}$$

where $h_0, \ldots, h_n$ are closed proper convex functions in $\Re^m$ ($m \geq 1$). We can also allow for linear equality constraints in the above problem but, for simplicity, we do not consider this more general case.

For any convex function g, we denote by dom g the effective domain of g, i.e., dom $g = \{ z \mid g(z) < +\infty \}$. For any convex set C, we denote by int(C) and ri(C), respectively, the interior and the relative interior of C. We make the following standing assumptions regarding the convex program (26).

Assumption C. (a) There exists a $\bar{z} \in$ ri(dom h_0) satisfying $h_i(\bar{z}) \leq 0$ for all i, with strict inequality holding whenever h_i is not affine.

(b) The level sets of the program (26), namely, sets of the form

$$\{ z \mid h_0(z) \leq \xi, h_1(z) \leq 0, \ldots, h_n(z) \leq 0 \}$$

with $\xi \in \Re$, are bounded.

Note that by part (a) of Assumption C, program (26) has at least one feasible solution. This, together with part (b) of Assumption C, implies that the set of optimal solutions for (26) is nonempty and compact.

By associating a Lagrange multiplier x_i with the constraint $h_i(z) \leq 0$ for every i, we obtain the following dual function:

$$f(x) = \begin{cases} \sup_z \{-\langle x, h(z)\rangle - h_0(z)\} & \text{if } x \geq 0, \\ +\infty & \text{otherwise,} \end{cases} \tag{27}$$

where we denote by $h(z)$ the vector in $\Re^n$ whose ith component is $h_i(z)$ and by x the vector in $\Re^n$ whose ith component is x_i. It is well known that f is a closed proper convex function.

It is known that when Assumption C holds, the set of Kuhn–Tucker vectors for the convex program (26) is nonempty and equals the set of minimizers of f (see [Roc70, Thm. 28.2]). Moreover, strong duality holds in the sense that the optimal value of problem (26) equals the negative of the minimum value of f. Thus, we can consider solving problem (26) by minimizing the dual function f of (27) and, for this purpose, we can use the parallel PPM algorithm (9)–(11). We show below that, for $\phi(\cdot) = \frac{1}{2}|\cdot|^2$, the proximal minimization step (9) in this algorithm is well defined and can be implemented with the use of quadratic augmented Lagrangian functions. Fix any nonempty subset I of $\{1, \ldots, n\}$, any $x \in \Re^n$, and any scalar $c > 0$. Consider the following convex program associated with I, x and c:

$$\text{minimize } h_0(z) + \frac{1}{2c} \sum_{i \in I} [x_i + c h_i(z)]_+^2 \tag{28a}$$

$$\text{s.t. } h_i(z) \leq 0, \qquad i \notin I, \tag{28b}$$

where, for any number a, we denote by $[a]_+$ the positive part of a, i.e., $[a]_+ = \max\{0, a\}$. This program has at least one feasible solution (namely, $\bar{z}$) and its level sets are bounded (since any direction of unboundedness for this program would also be a direction of unboundedness for the program (26)), so it has at least one optimal solution. Let z' be any such optimal solution. Notice that the program (28) has a feasible solution (namely, $\bar{z}$), which is in the relative interior of the effective domain of its cost function and satisfies with strict inequality all constraints for which h_i is not affine. Then, by Theorem 28.2 in [Roc70], the program (28) has a Kuhn–Tucker vector. Fix any such Kuhn–Tucker vector and let x_i', $i \notin I$, denote its component associated with the constraint $h_i(z) \leq 0$. Let x' be the vector in $\Re^n$ whose ith component is x_i' for all $i \notin I$ and, otherwise, is

$$x_i' = [x_i + c h_i(z')]_+ \quad \forall\, i \in I. \tag{29}$$

We claim that x' is a minimizer of the function

$$y \mapsto f(y) + \frac{1}{2c} \sum_{i \in I} |y_i - x_i|^2. \tag{30}$$

To see this, notice from the Kuhn–Tucker conditions for program (28) that

$$x_i' = [x_i' + ch_i(z')]_+ \qquad \forall\, i \notin I \tag{31}$$

and

$$0 \in \partial h_0(z') + \sum_{i \in I} [x_i + ch_i(z')]_+ \partial h_i(z') + \sum_{i \notin I} x_i' \partial h_i(z').$$

This equation together with (29) yields

$$0 \in \partial h_0(z') + \sum_{i=1}^{n} x_i' \partial h_i(z'),$$

implying

$$z' = \arg\min_z \{\langle x', h(z)\rangle + h_0(z)\}. \tag{32}$$

Let us write (29) and (31) equivalently as

$$0 \in T_i - h_i(z') + \frac{x_i' - x_i}{c} \quad \forall\, i \in I, \qquad 0 \in T_i - h_i(z') \quad \forall\, i \notin I,$$

where T_i is the interval $[0, +\infty)$ if $x_i' = 0$ and otherwise is just the origin $\{0\}$. From (32) and the definition of f (cf. (27)), we see that the Cartesian product $(T_1 - h_1(z')) \times \cdots \times (T_n - h_n(z'))$ is precisely $\partial f(x')$. This together with the above relation shows that x' is a minimizer of the function given by (30).

The above discussion shows that the parallel PPM algorithm with quadratic proximal term, applied to minimizing the dual function f of (27), is well defined and that each partial proximal minimization (9) can be achieved by solving a convex program of the form (28). A key feature of the program (28) is that only a subset of the constraints are eliminated. By carefully choosing the subsets to eliminate, one can preserve special structures of the cost function and perhaps also attain a faster rate of convergence; see [Ber82, §2.4], [Dun89], and [Alj90] for discussions of augmented Lagrangian methods of this type.

The above dual application of the parallel PPM algorithm is closely related to the parallel constraint distribution algorithm of Ferris and Mangasarian [FeM91]. In particular, by noting that the program (28) is identical in form to that appearing in Theorem 3.2 in [FeM91], we see that the two algorithms differ only in that the latter requires the subsets I effectively used at each iteration to form a partition of $\{1, \ldots, n\}$ and that, instead of taking a convex combination of the $\tilde{x}_I^k$'s, it extracts the coordinates indexed by I from $\tilde{x}_I^k$ to form x^{k+1}. Thus, the parallel PPM algorithm updates in a manner reminiscent of Cimmino's method [Cim38], while the parallel constraint distribution algorithm updates in a manner reminiscent of a Jacobi method. The subsequent paper [Fer91a] uses updates similar to the ones of the present paper and presents computational results showing an improved performance over the algorithm of [FeM91].

Under a strong regularity assumption that guarantees boundedness of the set of Kuhn–Tucker vectors for the convex program (26), we immediately obtain as a consequence of Proposition 2(a) the following convergence result for the parallel PPM algorithm.

PROPOSITION 5. *Assume that there is a point in* dom h_0 *satisfying all the constraints in* (26b) *with strict inequality. Consider the parallel* PPM *algorithm* (9)–(11) *with* $\{c^k\}$ *monotonically nondecreasing and with* $\phi(\cdot) = \frac{1}{2}|\cdot|^2$, *applied to minimize* f *given by* (27). *Then a sequence* $\{x^k\}$ *generated by the algorithm is bounded, each of its cluster points is a minimizer of* f, *and* $\{-f(x^k)\}$ *converges to the optimal value of the program* (26).

Proof. By the given hypothesis, the convex program (26) is strictly consistent in the terminology of [Roc70, p. 300]. Since the optimal value of problem (26) is finite, it follows from Corollary 29.1.5 in [Roc70] that the Kuhn–Tucker vectors for problem (26) form a nonempty compact convex subset of $\Re^n$. Since these Kuhn–Tucker vectors are precisely the minimizers of f, the hypothesis of Proposition 2(a) holds, and the result follows from that proposition. □

By translating the growth conditions (15) and (24) on f into conditions on h_0 and $h_1, \ldots, h_m$, and then applying Propositions 3 and 4, we analogously obtain the following two rate of convergence results for the parallel PPM algorithm.

PROPOSITION 6. *Assume that there is a point in* dom h_0 *satisfying all the constraints in* (26b) *with strict inequality. Furthermore, assume that there exist scalars* $\alpha > 1$, $\beta > 0$, *and* $\delta > 0$ *such that*

$$p(u) - p(0) - \langle u, x^* \rangle \leq (\alpha - 1)\beta \left(\frac{||u||}{\alpha\beta} \right)^{\alpha/(\alpha-1)} \quad \forall\, x^* \in \partial p(0) \text{ and } u \text{ with } ||u|| \leq \alpha\beta\delta^{\alpha-1},$$

where $p : \Re^n \to (-\infty, +\infty]$ *is the perturbation function given by*

$$p(u) = \min\{\, h_0(z) \mid h(z) \leq u \,\}.$$

Let $\{x^k\}$ *be a sequence generated by the parallel* PPM *algorithm* (9)–(11) *with* $\{c^k\}$ *monotonically nondecreasing and with* $\phi(\cdot) = \frac{1}{2}|\cdot|^2$, *applied to minimize* f *given by* (27). *Let* $f^* = \min_x f(x) = -p(0)$.

(a) *If* $1 < \alpha < 2$, *then* $\{f(x^k)\}$ *converges to* f^* *superlinearly with order* $2/\alpha$.

(b) *If* $\alpha = 2$ *and* $c^k \to \infty$, *then* $\{f(x^k)\}$ *converges to* f^* *superlinearly.*

(c) *If* $\alpha = 2$ *and* $c^k \to \bar{c} < \infty$, *then* $\{f(x^k)\}$ *converges to* f^* *R-linearly.*

Proof. As was shown in the proof of Proposition 5, the set X^* is nonempty and compact. We show below that f satisfies (15) with α, β, δ as given, so the claim immediately follows from Proposition 3(b), (c).

Fix any $x \in \Re^n$ with $\rho(x; X^*) \leq \delta$. First assume that $x \notin X^*$. Let x^* be the minimizer of f nearest to x, i.e., $\rho(x; X^*) = ||x - x^*||$. Also let

$$u = \alpha\beta(x - x^*)||x - x^*||^{\alpha-2}.$$

It is well known that p is the conjugate function of f, so that, by [Roc70, Thm. 23.5],

$$x^* \in \partial p(0).$$

Also, direct calculation finds that $||u|| \leq \alpha\beta\delta^{\alpha-1}$ and

$$(\alpha - 1)\beta \left(\frac{||u||}{\alpha\beta} \right)^{\alpha/(\alpha-1)} + \langle u, x^* \rangle = \langle u, x \rangle - \beta||x - x^*||^\alpha.$$

Thus, the hypothesis on p yields

$$p(u) - p(0) \leq \langle u, x \rangle - \beta||x - x^*||^\alpha.$$

Also, we have

$$p(u) = \sup_y \{\langle u, y\rangle - f(y)\} \geq \langle u, x\rangle - f(x), \qquad p(0) = -f^*,$$

which together with the above inequality yields

$$-f(x) + f^* \leq -\beta||x - x^*||^\alpha.$$

Rearranging terms and using $\rho(x; X^*) = ||x - x^*||$, we obtain

$$\beta\rho(x; X^*)^\alpha \leq f(x) - f^*.$$

Now assume that $x \in X^*$. Then the above relation holds trivially. □

As is shown by the preceding proof, the growth condition in Proposition 6 can alternatively be replaced by the growth condition (15) on the dual function f. Depending on the problem structure, one condition may be easier to verify than the other.

PROPOSITION 7. *Assume that the cost function h_0 is the sum of the indicator function of a polyhedral set and a strongly convex differentiable function whose gradient is Lipschitz continuous everywhere. Also assume that the constraint functions $h_1, \ldots, h_n$ are affine. Let $\{x^k\}$ be a sequence generated by the parallel* PPM *algorithm* (9)–(11) *with $\{c^k\}$ monotonically nondecreasing and with $\phi(\cdot) = \frac{1}{2}|\cdot|^2$, applied to minimize the dual function f given by* (27). *Then $\{-f(x^k)\}$ converges R-linearly to the optimal value of the convex program* (26).

Proof. It can be seen that, in this case, the convex program (26) is a special case of the convex program (2.2) studied in [LuT93] and that Assumptions A and B therein hold. Then, by Theorem 4.1 in [LuT93], f satisfies Assumption B when restricted to the level set $\{ x \mid x \geq 0,\ f(x) \leq f(x^0) \}$. Since, by Proposition 2(b), $\{f(x^k)\}$ is monotonically nonincreasing so the sequence $\{x^k\}$ lies in this level set, we can invoke Proposition 4 to conclude that $\{f(x^k)\}$ converges to $\min_x f(x)$ R-linearly. □

We remark that Proposition 7 is similar to the rate of convergence results obtained in [FeM91] and [Fer91a], but these references treat only the case where h_0 is strongly convex quadratic and $h_1, \ldots, h_n$ are affine.

REFERENCES

[Alj90] M. ALJAZZAF, *Multiplier Methods with Partial Elimination of Constraints for Nonlinear Programming*, Ph.D. thesis, Dept. of Mathematics, North Carolina State University, Raleigh, 1990.

[BeT89] D. P. BERTSEKAS AND J. N. TSITSIKLIS, *Parallel and Distributed Computation: Numerical Methods*, Prentice-Hall, Englewood Cliffs, NJ, 1989.

[Ber75] D. P. BERTSEKAS, *Necessary and sufficient conditions for a penalty method to be exact*, Math. Programming, 9 (1975), pp. 87–99.

[Ber82] ———, *Constrained Optimization and Lagrange Multiplier Methods*, Academic Press, New York, 1982.

[CeZ92] Y. CENSOR AND S. A. ZENIOS, *Proximal minimization algorithm with D-functions*, J. Optim. Theory Appl., 73 (1992), pp. 451–463.

[ChT93] G. CHEN AND M. TEBOULLE, *Convergence analysis of a proximal-like minimization algorithm using Bregman functions*, SIAM J. Optim., 3 (1993), pp. 538–543.

[Cim38] G. CIMMINO, *Calcolo Approssimato per le Soluzioni dei Sistemi di Equazioni Lineari*, Ricerca Scientifica, XVI Seria II, Anno IX, 1 (1938), pp. 326–333.

[Dun89] J. C. DUNN, *Formal augmented Newtonian projection methods for continuous-time optimal control*, Proc. 28th IEEE Conf. Decision and Control, Tampa, FL, 1989.

[EcB92] J. ECKSTEIN AND D. P. BERTSEKAS, *On the Douglas–Rachford splitting method and the proximal point algorithm for maximal monotone operators*, Math. Programming, 55 (1992), pp. 263–275.

[Eck93] J. ECKSTEIN, *Nonlinear proximal point algorithms using Bregman functions, with applications to convex programming*, Math. Oper. Res., 18 (1993), pp. 202–226.

[FeM91] M. C. FERRIS AND O. L. MANGASARIAN, *Parallel constraint distribution*, SIAM J. Optim., 1 (1991), pp. 487–500.

[Fer91a] M. C. FERRIS, *Parallel Constraint Distribution in Convex Quadratic Programming*, Department of Computer Science Report, University of Wisconsin, Madison, WI, 1991; Math. Oper. Res., to appear.

[Fer91b] ———, *Finite termination of the proximal point algorithm*, Math. Programming, 50 (1991), pp. 359–366.

[GaB84] E. M. GAFNI AND D. P. BERTSEKAS, *Two-metric projection methods for constrained optimization*, SIAM J. Control Optim., 22 (1984), pp. 936–964.

[GoT79] E. G. GOLSHTEIN AND N. V. TRETJIAKOV, *Modified Lagrangians in convex programming and their generalizations*, Math. Programming Stud., 10 (1979), pp. 86–97.

[Gul91] O. GÜLER, *On the convergence of the proximal point algorithm for convex minimization*, SIAM J. Control Optim., 29 (1991), pp. 403–419.

[Ha90] C. D. HA, *A generalization of the proximal point algorithm*, SIAM J. Control Optim., 28 (1990), pp. 503–512.

[KoB72] B. W. KORT AND D. P. BERTSEKAS, *A new penalty function method for constrained minimization*, Proc. 1971 IEEE Conf. Decision and Control, New Orleans, LA, 1972.

[KoB76] ———, *Combined primal-dual and penalty methods for convex programming*, SIAM J. Control Optim., 14 (1976), pp. 268–294.

[Luq84] F. J. LUQUE, *Nonlinear Proximal Point Algorithms*, Ph.D. thesis, Dept. of Civil Engineering and Operations Research Center, Massachusetts Institute of Technology, Cambridge, MA, 1984.

[Luq86] F. J. LUQUE, *The Nonlinear Proximal Point Algorithm*, Laboratory for Information and Decision Systems Report P-1598, Massachusetts Institute of Technology, Cambridge, MA, 1986.

[LuT93] Z. Q. LUO AND P. TSENG, *On the Convergence Rate of Dual Ascent Methods for Linearly Constrained Convex Minimization*, Math. Oper. Res., 18 (1993), pp. 846–867.

[LuT92] ———, *On the linear convergence of descent methods for convex essentially smooth minimization*, SIAM J. Control Optim., 30 (1992), pp. 408–425.

[Mar70] B. MARTINET, *Regularisation d'inéquations variationelles par approximations successives*, Revue Française d'Automatique et Informatique Recherche Opérationelle, 4 (1970), pp. 154–159.

[Mar72] ———, *Determination approchée d'un point fixe d'une application pseudo-contractante*, Compte Rendu Académie des Sciences de Paris, 274 (1972), pp. 163–165.

[OrR70] J. M. ORTEGA AND W. C. RHEINBOLDT, *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, New York, 1970.

[Pan87] J.-S. PANG, *A posteriori error bounds for the linearly-constrained variational inequality problem*, Math. Oper. Res., 12 (1987), pp. 474–484.

[Roc70] R. T. ROCKAFELLAR, *Convex Analysis*, Princeton University Press, Princeton, NJ, 1970.

[Roc73] ———, *A dual approach to solving nonlinear programming problems by unconstrained minimization*, Math. Programming, 5 (1973), pp. 354–373.

[Roc76] ———, *Monotone operators and the proximal point algorithm*, SIAM J. Control Optim., 14 (1976), pp. 877–898.

[TsB93] P. TSENG AND D. P. BERTSEKAS, *On the convergence of the exponential multiplier method for convex programming*, Math. Programming, 60 (1993), pp. 1–19.

SIAM J. OPTIMIZATION
Vol. 4, No. 3, pp. 573–595, August 1994
© 1994 Society for Industrial and Applied Mathematics
008

EXPOSING CONSTRAINTS*

JAMES V. BURKE† AND JORGE J. MORÉ‡

Abstract. The development of algorithms and software for the solution of large-scale optimization problems has been the main motivation behind the research on the identification properties of optimization algorithms. The aim of an identification result for a linearly constrained problem is to show that if the sequence generated by an optimization algorithm converges to a stationary point, then there is a nontrivial face F of the feasible set such that after a finite number of iterations, the iterates enter and remain in the face F. This paper develops the identification properties of linearly constrained optimization algorithms without any nondegeneracy or linear independence assumptions. The main result shows that the projected gradient converges to zero if and only if the iterates enter and remain in the face exposed by the negative gradient. This result generalizes results of Burke and Moré obtained for nondegenerate cases.

Key words. nondegeneracy, strict complementarity, projected gradient, active constraints, linearly constrained problems

AMS subject classifications. 65K05, 90C20, 90C25, 90C30

1. Introduction.

The development of algorithms and software for the solution of large-scale constrained optimization problems,

$$\min\{f(x) : x \in \Omega\}, \tag{1.1}$$

has been the main motivation behind the research on the identification properties of optimization algorithms. Much of this research has been done under the assumption that the set Ω is a general convex set and that the stationary points of the optimization problem (1.1) are nondegenerate. In this work we show that it is possible to develop the identification properties of linearly constrained optimization algorithms without any nondegeneracy or linear independence assumptions.

The aim of an identification result is to show that if $\{x_k\}$ is a sequence in Ω that converges to a stationary point x^*, then there is an index $k_0 > 0$ and a nontrivial face $F(x^*)$ of Ω with $x_k \in F(x^*)$ for all $k \geq k_0$. These results are important because they show that eventually the behavior of the algorithm is determined by the properties of f on the face $F(x^*)$. For recent work on the identification properties of optimization algorithms, see, for example, Conn, Gould, and Toint [7], Dunn [11], Wright [26], Burke [3], Burke, Moré, and Toraldo [5], Lescrenier [17], Kelley and Sachs [16], and Wright [27].

In linearly constrained problems an identification result can also be expressed in terms of the indices of the active constraints. For example, if Ω has the specific representation

$$\Omega = \{x \in \mathbb{R}^n : \langle c_j, x\rangle \geq \delta_j,\ 1 \leq j \leq m\}, \tag{1.2}$$

for some vectors $c_j \in \mathbb{R}^n$ and scalars δ_j, then all faces F of Ω are of the form

$$F = \{x \in \Omega : \langle c_j, x\rangle = \delta_j,\ j \in \mathcal{A}(F)\}$$

* Received by the editors June 25, 1992; accepted for publication June 30, 1993.

† Department of Mathematics, University of Washington, Seattle, Washington 98195. The work of this author was supported in part by National Science Foundation grant DMS-9102059.

‡ Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, Illinois 60439. The work of this author was supported by the Office of Scientific Computing, U.S. Department of Energy, contract W-31-109-Eng-38 (more@mcs.anl.gov).

for some index set $\mathcal{A}(F)$. Thus, identification results can be given in terms of an index set that defines the face $F(x^*)$. In this paper we obtain results in terms of the face structure of Ω; results in terms of index sets of active constraints are direct consequences of these results.

An important consequence of the assumption that the stationary point x^* is nondegenerate is that $F(x^*)$ is the unique face of Ω that contains x^* in the relative interior. Thus, it follows that $x_k \in \operatorname{ri}\{F(x^*)\}$ for all $k \geq k_0$, where $\operatorname{ri}\{\cdot\}$ denotes the relative interior. One of the main difficulties in extending identification results to the degenerate case is that there may be no face $F(x^*)$ such that $x_k \in \operatorname{ri}\{F(x^*)\}$ for all k sufficiently large. However, we show that there is a face $F(x^*)$ such that $x_k \in F(x^*)$ for all k sufficiently large.

The approach in this paper is reminiscent of the approach of Burke and Moré in [4]. In particular, the approach depends on the concept of an exposed face, and on the facial geometry of a convex set Ω. These results are developed in §§2 and 3. The definition of an exposed face and several important properties of exposed faces are presented in §2, while §3 contains a key result on the existence of a partition of $\mathbb{R}^n$ based on the face structure of Ω.

The main result of this paper is found in §4. This result is a characterization of the identification properties of a sequence $\{x_k\}$ in terms of the projected gradient and the face $E[-\nabla f(x^*)]$ of Ω exposed by $-\nabla f(x^*)$. If $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable on the polyhedral set Ω, and $\{x_k\}$ is a sequence in Ω that converges to a stationary point x^* of (1.1), then we show that

$$\lim_{k\to\infty} P_{T(x_k)}\left[-\nabla f(x_k)\right] = 0 \tag{1.3}$$

if and only if there is a $k_0 > 0$ such that

$$x_k \in E[-\nabla f(x^*)], \qquad k \geq k_0. \tag{1.4}$$

This result has important ramifications because optimization algorithms for linearly constrained problems tend to satisfy (1.3). This can be seen by noting that if the polyhedral set Ω has the representation (1.2), then the projected gradient is

$$P_{T(x)}\left[-\nabla f(x)\right] = -\nabla f(x) + \sum_{j\in\mathcal{A}(x)} \lambda_j c_j,$$

where $\mathcal{A}(x)$ is the set of active constraints at $x \in \Omega$, and λ_j is a nonnegative estimate (a precise definition is given in §6) of the Lagrange multiplier for the ith constraint.

The identification property (1.4) can be expressed in terms of Lagrange multipliers at the solution because we show that $E[-\nabla f(x^*)]$ is defined in terms of the active set by

$$x \in E[-\nabla f(x^*)] \iff \{i \in \mathcal{A}(x^*) : \lambda_i^* > 0\} \subset \mathcal{A}(x), \qquad x \in \Omega,$$

where λ_i^* is the Lagrange multiplier for the ith constraint. This result depends on the choice of Lagrange multipliers.

The identification property (1.4) has an equivalent formulation that is clearly independent of the choice of Lagrange multipliers. Section 5 shows that it is possible to define a set of strictly binding constraints $\mathcal{B}_s^*$ at a stationary point x^* independent of the choice of Lagrange multipliers, and that

$$x \in E[-\nabla f(x^*)] \iff \mathcal{B}_s^* \subset \mathcal{A}(x), \qquad x \in \Omega.$$

This leads, in particular, to a version of the results of §4 in terms of $\mathcal{B}_s^*$.

Section 4 also contains a discussion of the connection of these results to the identification results of Burke and Moré [4], and to the convergence results for the class of trust region methods for bound constrained optimization problems proposed by Conn, Gould, and Toint [7], [8]. The connection between these results and the class of trust region methods for general linearly constrained methods analyzed by Moré [18] and Burke, Moré, and Toraldo [5] will be reported in a later paper.

We end the paper with two applications of these identification results. Section 6 examines the influence of degeneracy on the standard second-order sufficiency conditions. In particular, we show that if x^* is degenerate, a satisfactory convergence analysis can be obtained if we assume that $\nabla^2 f(x^*)$ is positive definite on the affine hull of $E[-\nabla f(x^*)] - x^*$. Section 7 examines the implication of the identification results to the GPCG algorithm of Moré and Toraldo [19]. In particular, we show that the convergence condition of the GPCG algorithm is satisfied in a finite number of iterations even when the solution is degenerate.

2. Exposed faces. The geometric approach to the identification properties of optimization algorithms requires an understanding of the face structure of a convex set Ω. In this section we provide some of the necessary results and background from convex analysis.

Recall that the *affine hull* $\operatorname{aff}\{\Omega\}$ of a convex set Ω in $\mathbb{R}^n$ is the smallest affine set that contains Ω, and the *relative interior* $\operatorname{ri}(\Omega)$ of Ω is the interior of Ω relative to $\operatorname{aff}\{\Omega\}$. In all cases we assume that Ω is not empty.

A nonempty subset F of a convex set Ω is a *face* of Ω if every convex subset of Ω whose relative interior meets F is contained in F. Thus, if x and y are in Ω and $\lambda x + (1-\lambda)y$ lies in F for some λ in $(0,1)$, then both x and y must belong to F. A basic result on the face structure of a convex set is that the relative interiors of the faces of Ω form a partition of Ω. For future reference, we state this result formally.

THEOREM 2.1. *If $\mathcal{F}$ is the collection of all faces of the convex set Ω, then the collection*

$$\{\operatorname{ri}(F) : F \in \mathcal{F}\}$$

is a partition of Ω.

This result can be found, for example, in Rockafellar [24, Thm. 18.2]. Note that this result shows that every point $x \in \Omega$ can be associated with a unique face $F(x)$ of Ω such that $x \in \operatorname{ri}(F)$.

We are concerned with faces of Ω that are exposed by a given vector d in $\mathbb{R}^n$. A nonempty subset F of a convex set Ω is *exposed* by a vector $d \in \mathbb{R}^n$ if $F = E(d)$, where

$$E(d) = \arg\max\{\langle d, x\rangle : x \in \Omega\}$$

for some inner product $\langle \cdot, \cdot \rangle$ of $\mathbb{R}^n$. A computation shows that $E(d)$ is a face whenever $E(d)$ is not empty.

Every face of a polyhedron is exposed (see, for example, Theorem 2.4.12 of Stoer and Witzgall [25]), but this is not the case for general convex sets. For example, in the convex set Ω defined by

$$\Omega = \left\{(\xi_1, \xi_2) : -1 \le \xi_2 \le (1 - \xi_1^2)^{1/2},\ 0 \le \xi_1 \le 1\right\}, \tag{2.1}$$

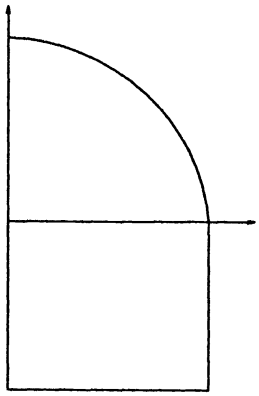


FIG. 2.1. *Convex set defined by* (2.1).

the point $(1,0)$ is a face of Ω that is not exposed by any vector. This can be seen clearly in Fig. 2.1.

The concept of an exposed face is closely related to the concept of a normal cone. For a convex set Ω, the normal cone at x in Ω is defined by

$$N(x) = \{u \in \mathbb{R}^n : \langle u, y - x\rangle \leq 0,\ y \in \Omega\}.$$

The tangent cone $T(x)$ is the dual of the normal cone. Thus $v \in T(x)$ if and only if $\langle v, u\rangle \leq 0$ for all $u \in N(x)$. The tangent cone $T(x)$ can also be defined as the closure of all vectors $v \in \mathbb{R}^n$ such that $x + \alpha v \in \Omega$ for all $\alpha > 0$ sufficiently small. Figure 2.1 can be used to illustrate these definitions. For example,

$$N(1, \xi_2) = \{v \in \mathbb{R}^2 : v_1 = 0\}, \qquad T(1, \xi_2) = \{v \in \mathbb{R}^2 : v_1 \leq 0\}$$

for $-1 < \xi_2 \leq 0$.

Exposed faces and normal cones are related by the observation that $x \in E(d)$ if and only if $d \in N(x)$. We make use of this observation throughout this section.

The concept of an exposed face is also related to the standard first-order optimality conditions for problem (1.1). Recall that the standard first-order conditions for a stationary point x^* of problem (1.1) are that

$$\langle \nabla f(x^*), x - x^*\rangle \geq 0, \qquad x \in \Omega.$$

In terms of normal cones, this is equivalent to requiring that

$$-\nabla f(x^*) \in N(x^*).$$

The following result shows the connection between stationary points and exposed faces.

LEMMA 2.2. *If Ω in $\mathbb{R}^n$ is a closed convex set, then x^* is a stationary point for problem* (1.1) *if and only if $x^* \in E[-\nabla f(x^*)]$.*

The proof of this result is a direct consequence of the observation that $x \in E(d)$ if and only if $d \in N(x)$.

Lemma 2.2 shows that any stationary point x^* belongs to the exposed face $E[-\nabla f(x^*)]$. The stationary point x^* may belong to other faces of Ω, but in §4 we show that optimization algorithms tend to generate sequences $\{x_k\}$ with $x_k \in$

$E[-\nabla f(x^*)]$ for all k sufficiently large. In this section we develop part of the necessary machinery for this result.

We extend the concept of normal and tangent cones at a point $x \in \Omega$ by defining a normal cone $N(F)$ and a tangent cone $T(F)$ for each $F \in \mathcal{F}$ by

$$N(F) \equiv N(x), \quad T(F) \equiv T(x), \quad x \in \operatorname{ri}(F).$$

This definition relies on the result of Burke and Moré [4] that normal and tangent cones are independent of $x \in \operatorname{ri}(F)$ for any $F \in \mathcal{F}$.

The intersection of an arbitrary collection of faces is easily shown to be a face, but the union of faces is not necessarily a face. On the other hand, the following result shows that the union of all faces $F \in \mathcal{F}$ such that $d \in N(F)$ is the face exposed by d.

LEMMA 2.3. *If Ω in $\mathbb{R}^n$ is a closed convex set, then*

$$E(d) = \bigcup_{F \in \mathcal{F}(d)} F$$

for any $d \in \mathbb{R}^n$, where $\mathcal{F}(d)$ is the collection of faces $F \in \mathcal{F}$ such that $d \in N(F)$.

Proof. Assume first that $x \in E(d)$. Then $d \in N(x)$, and since Theorem 2.1 guarantees that $x \in \operatorname{ri}(F)$ for some $F \in \mathcal{F}$, we have $d \in N(x) = N(F)$. This shows that

$$E(d) \subset \bigcup_{F \in \mathcal{F}(d)} F.$$

We complete the proof by showing that if $F \in \mathcal{F}(d)$ then $F \subset E(d)$. Assume that $x \in F$ for some $F \in \mathcal{F}(d)$. We can choose a sequence $\{x_k\}$ with $x_k \in \operatorname{ri}(F)$ such that $\{x_k\}$ converges to x. Since $x_k \in \operatorname{ri}(F)$ and $d \in N(F)$, we have $d \in N(x_k)$, and hence, $x_k \in E(d)$. Since the exposed face $E(d)$ is closed and $\{x_k\}$ converges to x, we obtain that $x \in E(d)$ as desired. □

The proof of Lemma 2.3 yields the stronger result that $\{\operatorname{ri}(F) : F \in \mathcal{F}(d)\}$ is a partition of $E(d)$. This result, however, is not needed in this paper.

Lemma 2.3 also shows that if $d \in N(F)$ for some $F \in \mathcal{F}$, then $F \subset E(d)$. For a general convex set Ω, it is difficult to characterize the vectors $d \in N(F)$ that expose a face F. On the other hand, the following result of Burke [3, Thm. 4.1] provides a complete characterization for polyhedral Ω.

THEOREM 2.4. *Let Ω be a polyhedral set in $\mathbb{R}^n$.*

If $x \in \Omega$, then $x \in \operatorname{ri}\{E(d)\}$ if and only if $d \in \operatorname{ri}\{N(x)\}$.

If $F \in \mathcal{F}$, then $F = E(d)$ if and only if $d \in \operatorname{ri}\{N(F)\}$.

Burke [3] proved only the first claim in Theorem 2.4, but it is not difficult to show that both claims are equivalent.

Theorem 2.4 fails if Ω is a general convex set. For example, if Ω is the set in Fig. 2.1, and $F = \{(1,0)\}$, then F is not exposed by $d = (1,0)$, but $d \in \operatorname{ri}\{N(F)\}$. Moreover, if $F = E(d)$ for $d = (0,1)$, then d exposes F, but $d \notin \operatorname{ri}\{N(F)\}$. This example shows that the assumption of polyhedrality is required in Theorem A.1 of Burke [3].

The above example also shows that a face may not be exposed by a vector d in $\operatorname{ri}\{N(F)\}$. In the remainder of this section we show that F is exposed by a vector $d \in \operatorname{ri}\{N(F)\}$ if F is a quasi-polyhedral face. This result is interesting because it leads to a connection with the identification results of Burke and Moré [4] for general

convex sets. However, note that the remainder of this section is not needed for the main results in this paper.

Burke and Moré [4] defined quasi-polyhedral faces in terms of the lineality of the tangent cone $T(x)$. Recall that for a cone K in $\mathbb{R}^n$, the lineality $\operatorname{lin}\{K\}$ of the cone K is the largest subspace contained in K. Hence, $\operatorname{lin}\{K\} = K \cap (-K)$.

DEFINITION. *A face F of a convex set Ω is quasi-polyhedral if*

$$\operatorname{aff}\{F\} = x + \operatorname{lin}\{T(x)\}, \qquad x \in \operatorname{ri}(F).$$

The convex set Ω in Fig. 2.1 can be used to illustrate the difference between exposed faces and quasi-polyhedral faces. This set has an infinite number of faces, but the only face that is not exposed by a vector $d \in \operatorname{ri}\{N(F)\}$ is $F = \{(0,1)\}$. Also note that Ω has six quasi-polyhedral faces and that these faces are exposed by any vector $d \in \operatorname{ri}\{N(F)\}$.

Burke and Moré [4] show that the lineality of the tangent cone is the orthogonal complement of the normal cone; that is,

$$\operatorname{lin}\{T(x)\} = N(x)^{\perp},$$

where the orthogonal complement $S^{\perp}$ of a set S is the subspace of vectors v such that $\langle v, w\rangle = 0$ for all $w \in S$. This implies that

$$\operatorname{lin}\{T(x)\}^{\perp} = \operatorname{aff}\{N(x)\}.$$

These results lead to the decomposition

$$T(x) = \operatorname{lin}\{T(x)\} \oplus \operatorname{lin}\{T(x)\}^{\perp} \cap T(x) \tag{2.2}$$

of the tangent cone. This result is valid if $T(x)$ is replaced by a general convex cone K. See, for example, Stoer and Witzgall [25, Thm. 2.10.5].

THEOREM 2.5. *A quasi-polyhedral face F of a convex set Ω is exposed by any $d \in \operatorname{ri}\{N(F)\}$.*

Proof. The bulk of the proof consists of proving that if $x \in \Omega$, then

$$E(d) = \Omega \cap [x + \operatorname{lin}\{T(x)\}], \qquad d \in \operatorname{ri}\{N(x)\}. \tag{2.3}$$

Given this result, we complete the proof by noting that $F = \Omega \cap \operatorname{aff}\{F\}$ for any face of Ω, and thus

$$F = \Omega \cap [x + \operatorname{lin}\{T(x)\}], \qquad x \in \operatorname{ri}\{F\},$$

for a quasi-polyhedral face. Hence (2.3) yields that $F = E(d)$ if $d \in \operatorname{ri}\{N(x)\} = \operatorname{ri}\{N(F)\}$.

We now prove that (2.3) holds; we first show that $\Omega \cap [x + \operatorname{lin}\{T(x)\}]$ is a subset of $E(d)$. Choose any y in $\Omega \cap [x + \operatorname{lin}\{T(x)\}]$. Then $y - x \in \operatorname{lin}\{T(x)\} = N(x)^{\perp}$, and since $d \in N(x)$, we obtain that $\langle d, y - x\rangle = 0$. Since $y \in \Omega$, this implies that $y \in E(d)$.

We now show that $E(d)$ is a subset of $\Omega \cap [x + \operatorname{lin}\{T(x)\}]$. Choose any y in $E(d)$. Then $y \in \Omega$, and thus $y - x \in T(x)$. The decomposition (2.2) of the tangent cone $T(x)$ shows that we can write

$$y - x = v_1 + v_2, \quad v_1 \in \operatorname{lin}\{T(x)\}, \quad v_2 \in \operatorname{lin}\{T(x)\}^{\perp} \cap T(x).$$

If we show that $v_2 = 0$, then $y - x = v_1 \in \text{lin}\{T(x)\}$ as desired. We first show that $\langle d, v_2 \rangle = 0$. Note that $\langle d, y - x \rangle = 0$ because both y and x belong to $E(d)$, and that $\langle d, v_1 \rangle = 0$ because $d \in N(x)$ and $v_1 \in \text{lin}\{T(x)\} = N(x)^{\perp}$. Hence, $\langle d, v_2 \rangle = 0$.

We claim that $v_2 = 0$ since $d \in \text{ri}\{N(x)\}$, $v_2 \in \text{lin}\{T(x)\}^{\perp} = \text{aff}\{N(x)\}$, and $\langle d, v_2 \rangle = 0$. The proof of this claim is not difficult. Since $d \in \text{ri}\{N(x)\}$ and $v_2 \in \text{aff}\{N(x)\}$, we obtain that $d + \epsilon v_2 \in N(x)$ for $\epsilon > 0$ sufficiently small. We now use that $v_2 \in T(x)$ to obtain that $\langle d + \epsilon v_2, v_2 \rangle \leq 0$. Since $\langle d, v_2 \rangle = 0$, this implies that $v_2 = 0$, as desired. □

3. Face geometry. The main result of this section is an extension of Lemma 3.3 of Burke and Moré [4]. As we shall see, this extension is crucial to the results on exposing constraints. We present two different proofs of this result. The first proof is based on the following result of Robinson [22].

THEOREM 3.1. *Let Ω be a polyhedral set in $\mathbb{R}^n$. For any $d^* \in \mathbb{R}^n$ there is a neighborhood $S(d^*)$ of d^* such that*

$$E(d) \subset E(d^*), \qquad d \in S(d^*).$$

Proof. The proof is by contradiction. Assume, on the contrary, that there is a sequence $\{d_k\}$ that converges to d^*, and that $E(d_k)$ is not a subset of $E(d^*)$ for any $k \geq 0$. Since a polyhedral set has a finite number of faces (this result can be found, for example, in Rockafellar [24, Thm. 19.1]), there is a face E of Ω and a subsequence $\{d_{k_i}\}$ such that $E = E(d_{k_i})$.

We now show that $E \subset E(d^*)$. If $z \in E$, then $\langle d_{k_i}, z \rangle \geq \langle d_{k_i}, x \rangle$ for all $x \in \Omega$. The convergence of $\{d_k\}$ to d^* implies that $\langle d^*, z \rangle \geq \langle d^*, x \rangle$ for all $x \in \Omega$, and thus, $z \in E(d^*)$ as desired. This contradicts our assumption that $E(d_k)$ is not a subset of $E(d^*)$ for any $k \geq 0$. □

Robinson [22, Lemma 3.5] proves a stronger result than Theorem 3.1. However, we only need the above result to establish the main result of this section.

THEOREM 3.2. *Let Ω be a polyhedral set in $\mathbb{R}^n$. Assume that $\{x_k\}$ and $\{d_k\}$ are sequences in $\mathbb{R}^n$ with $x_k \in \Omega$ and $d_k \in N(x_k)$. If $\{x_k\}$ converges to x^* and $\{d_k\}$ converges to d^*, then $x_k \in E(d^*)$ for all k sufficiently large.*

Proof. This result is an immediate consequence of Theorem 3.1 because x_k is in $E(d_k)$. □

Theorem 3.2 fails if Ω is a general closed convex set even if we assume that $x^* \in \text{ri}\{F\}$ for a quasi-polyhedral face F. For example, consider the set Ω shown in Fig. 2.1. The sequence $\{x_k\}$ defined by

$$x_k = \left(\sin\left(\frac{\pi}{k}\right), \cos\left(\frac{\pi}{k}\right)\right)$$

belongs to Ω for $k > 1$, and $d_k = x_k \in N(x_k)$. However, $\{d_k\}$ converges to $d^* = (0, 1)$, and $E(d^*) = \{(0, 1)\}$. Thus, $x_k \notin E(d^*)$ for any $k > 1$.

Burke and Moré [4, Lemma 3.3] proved Theorem 3.2 for general Ω by assuming that $x^* \in \text{ri}\{F\}$ for a quasi-polyhedral face F and $d^* \in \text{ri}\{N(F)\}$. The assumption that $d^* \in \text{ri}\{N(F)\}$ is a nondegeneracy assumption that is avoided in Theorem 3.2 by assuming that Ω is polyhedral. In the nondegenerate case, the key to Theorem 3.2 is to show that

$$x^* + d^* \in \text{int}\{F + N(F)\},$$

where F is the unique face of Ω with $x^* \in \text{ri}(F)$. This result fails if we drop the assumption $d^* \in \text{ri}\{N(F)\}$. However, we claim that Theorem 3.2 can be obtained if

we show that

$$x^* + d^* \in \operatorname{int} \left\{ \bigcup_{F \in \mathcal{F}(d^*)} [F + N(F)] \right\}, \tag{3.1}$$

where $\mathcal{F}(d^*)$ is defined in Lemma 2.3 and $d^* \in N(x^*)$. In the remainder of this section we establish (3.1), and then show that (3.1) leads to another proof of Theorem 3.2.

Our development requires a few basic properties of projection operators. The projection $P : \mathbb{R}^n \to \Omega$ into a closed convex set Ω is defined by

$$P(x) = \operatorname{argmin} \{ \|y - x\| : y \in \Omega \},$$

where $\|\cdot\|$ is an inner-product norm. If the dependence of P on Ω is not clear from the context, we will use P_Ω to denote the projection into Ω.

The definition of the projection operator P can be characterized in terms of the inner product $\langle \cdot, \cdot \rangle$ by requiring that

$$\langle x - P(x), y - P(x) \rangle \le 0, \qquad y \in \Omega.$$

In terms of normal cones, this characterization requires that

$$x - P(x) \in N[P(x)]. \tag{3.2}$$

In particular, note that this characterization implies that

$$P(x + z) = x, \quad x \in \Omega, \quad z \in N(x). \tag{3.3}$$

We extend this result by proving that

$$P(x + z) = x, \quad x \in F, \quad z \in N(F) \tag{3.4}$$

for any face $F \in \mathcal{F}$. We prove (3.4) by noting that if $\{x_k\}$ is a sequence in $\operatorname{ri}(F)$ that converges to x, then $z \in N(F) = N(x_k)$, and thus (3.3) implies that $P(x_k + z) = x_k$. The result now follows from the continuity of the projection operator.

The proof of (3.1) requires the observation that the collection $\{F + N(F) : F \in \mathcal{F}\}$ covers $\mathbb{R}^n$. Note that $x = P(x) + [x - P(x)]$. Certainly $P(x) \in \operatorname{ri}(F)$ for some $F \in \mathcal{F}$, and thus the characterization (3.2) of the projection operator implies that

$$x - P(x) \in N[P(x)] = N(F).$$

Hence, $x = P(x) + [x - P(x)]$ belongs to $\operatorname{ri}(F) + N(F)$, and thus

$$\mathbb{R}^n = \bigcup_{F \in \mathcal{F}} [\operatorname{ri}(F) + N(F)].$$

The following result is not needed to establish (3.1), but is presented here because of the recent interest in partitions of $\mathbb{R}^n$ in terms of the faces of a convex set. See, for example, Goffin [15, Thm. 4.1] and Robinson [23, Prop. 2.2].

THEOREM 3.3. *If Ω is a polyhedral set in $\mathbb{R}^n$, then the collection*

$$\{F + \operatorname{ri}\{N(F)\} : F \in \mathcal{F}\}$$

forms a partition of $\mathbb{R}^n$.

Proof. If we define $F = E[x - P(x)]$, then Theorem 2.4 shows that $x - P(x)$ is in $\operatorname{ri}\{N(F)\}$. Moreover, since $x - P(x) \in N[P(x)]$, we obtain that $P(x) \in E[x - P(x)]$; that is, $P(x) \in F$. Hence,

$$x = P(x) + [x - P(x)] \in F + \operatorname{ri}\{N(F)\}.$$

This proves that

$$\mathbb{R}^n = \bigcup_{F \in \mathcal{F}} [F + \operatorname{ri}\{N(F)\}].$$

We now prove that this decomposition of $\mathbb{R}^n$ is a partition. Assume that $x_i \in F_i$ and $z_i \in \operatorname{ri}\{N(F_i)\}$ satisfy $x_1 + z_1 = x_2 + z_2$ for some F_1 and F_2 in $\mathcal{F}$. Then (3.3) implies that

$$x_1 = P(x_1 + z_1) = P(x_2 + z_2) = x_2.$$

Thus, $z_1 = z_2$. Since $z_i \in \operatorname{ri}\{N(F_i)\}$, Theorem 2.4 shows that $F_i = E(z_i)$, and thus $z_1 = z_2$ implies that $F_1 = F_2$ as desired. □

Theorem 3.3 does not hold for general convex sets Ω. For example, if Ω is the convex set shown in Fig. 2.1, then any point of the form $(0, \xi)$ with $\xi > 1$ does not belong to a set of the form $F + \operatorname{ri}\{N(F)\}$ for $F \in \mathcal{F}$.

We proved Theorem 3.3 because of our interest in partitions of $\mathbb{R}^n$ in terms of the faces of a convex set. The proof of (3.1), however, only needs the result that if Ω is poyhedral, then the collection $\{F + N(F) : F \in \mathcal{F}\}$ covers $\mathbb{R}^n$. Clearly, Theorem 3.3 yields this result.

We now prove (3.1). For this result we make use of the result that if Ω is polyhedral then Ω has a finite number of faces. This result can be found, for example, in Rockafellar [24, Thm. 19.1]. We also need to know that if Ω is closed and convex, then any face of Ω is closed. For this result, see Stoer and Witzgall [25, Thm. 3.6.6].

LEMMA 3.4. *Assume that Ω is a polyhedral set in $\mathbb{R}^n$. If $x \in \Omega$ and $d \in N(x)$, then*

$$x + d \in \operatorname{int}\{K(d)\},$$

where

$$K(d) = \bigcup_{F \in \mathcal{F}(d)} [F + N(F)],$$

and $\mathcal{F}(d)$ is defined in Lemma 2.3.

Proof. The first step in the proof is to show that if $F \in \mathcal{F}$, then $F + N(F)$ is closed. The definition of a normal cone shows that $N(F)$ is closed, and since Ω is closed, each face F is closed. Now let $\{y_k\}$ be a sequence in $F + N(F)$ that converges, and let $y_k = x_k + z_k$, where $x_k \in F$ and $z_k \in N(F)$. Since (3.4) implies that

$$x_k = P(x_k + z_k) = P(y_k),$$

and $\{y_k\}$ converges, it follows that $\{x_k\}$ converges. Moreover, since $y_k = x_k + z_k$, the sequence $\{z_k\}$ also converges. We now use that $x_k \in F$ and $z_k \in N(F)$, to conclude that $\{x_k\}$ converges to some $x \in F$ and $\{z_k\}$ converges to some $z \in N(F)$. Hence, $\{y_k\}$ converges to $x + z$ in $F + N(F)$. This shows that $F + N(F)$ is closed.

The next step in the proof is technical; we need to show that $x+d$ does not belong to the set

$$L(d) = \bigcup_{F \notin \mathcal{F}(d)} [F + N(F)].$$

Theorem 3.3 shows that $x+d \in F+N(F)$ for some $F \in \mathcal{F}$. Let $y \in F$ and $z \in N(F)$ be such that $x+d = y+z$. Then (3.4) implies that

$$x = P(x+d) = P(y+z) = y,$$

and thus $d = z \in N(F)$. Hence, $F \in \mathcal{F}(d)$. This shows that $x+d \notin L(d)$ as desired.

We have shown that $F+N(F)$ is closed for each $F \in \mathcal{F}$. Since Ω is polyhedral, there is a finite number of faces, and thus $L(d)$ is closed. Moreover, since $x+d \notin L(d)$, there is an open set U such that $x+d \in U$ and $U \cap L(d) = \emptyset$. Now note that Theorem 3.3 implies that $K(d) \cup L(d) = \mathbb{R}^n$, and thus $U \subset K(d)$. Hence,

$$x+d \in U \subset \operatorname{int}\{K(d)\},$$

as desired. □

Lemmas 2.3 and 3.4 can be used to give another proof of Theorem 3.2. Indeed, Lemma 3.4 implies that $x^* + d^* \in \operatorname{int}\{K(d^*)\}$ so that $x_k + d_k \in K(d^*)$ for all k sufficiently large. Since $d_k \in N(x_k)$,

$$x_k = P(x_k + d_k) \in P[K(d^*)]$$

by virtue of (3.3). Now note that (3.4) and Lemma 2.3 imply that

$$P[K(d^*)] \subset \left\{ \bigcup_{F \in \mathcal{F}(d^*)} P[F+N(F)] \right\} = \left\{ \bigcup_{F \in \mathcal{F}(d^*)} F \right\} = E(d^*),$$

and thus $x_k \in E(d^*)$ for all k sufficiently large.

4. Exposing constraints. Previous results on the identification properties of algorithms in the neighborhood of a stationary point either assumed linear independence of the active constraint normals or nondegeneracy of the stationary point. In this section we avoid these restrictions.

DEFINITION. *Given a convex set Ω, a stationary point x^* is nondegenerate if*

$$-\nabla f(x^*) \in \operatorname{ri}\{N(x^*)\}. \tag{4.1}$$

This nondegeneracy condition is due to Dunn [10]. An advantage of this definition is that it does not make any linear independence assumptions on the constraints. Also note that Burke and Moré [4] proved that if Ω is polyhedral, then x^* is nondegenerate if and only if there is a set of positive Lagrange multipliers. This result implies that this definition of nondegeneracy can be viewed as a generalization of the standard strict complementarity condition.

The main result of this section is an extension of the following identification result of Burke and Moré [4]. For this result recall that $F(x)$ is the unique face of Ω that contains x in the relative interior.

THEOREM 4.1. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable on the closed, convex set Ω. If $\{x_k\}$ is a sequence in Ω that converges to a nondegenerate stationary point x^* of (1.1), then*

$$\lim_{k\to\infty} P_{T(x_k)}\left[-\nabla f(x_k)\right] = 0 \tag{4.2}$$

if and only if there is a $k_0 > 0$ such that

$$x_k \in \operatorname{ri}\{F(x^*)\}, \qquad k \geq k_0.$$

An advantage of Theorem 4.1 is that it is independent of the representation of Ω. For applications, however, it is necessary to express Theorem 4.1 in terms of a specific representation of Ω. Note, in particular, that if Ω is the polyhedral set defined by the set of linear constraints

$$\Omega = \{x \in \mathbb{R}^n : \langle c_j, x\rangle \geq \delta_j,\ 1 \leq j \leq m\}, \tag{4.3}$$

for some vectors $c_j \in \mathbb{R}^n$ and scalars δ_j, then Calamai and Moré [6] proved that the projected gradient that appears in (4.2) can be expressed in the familiar form

$$P_{T(x)}\left[-\nabla f(x)\right] = -\nabla f(x) + \sum_{j\in\mathcal{A}(x)} \lambda_j c_j, \tag{4.4}$$

where the set of active constraints is defined by

$$\mathcal{A}(x) \equiv \{j : \langle c_j, x\rangle = \delta_j\},$$

and λ_j for $j \in \mathcal{A}(x)$ solves the bound constrained linear least squares problem

$$\min\left\{\left\|\nabla f(x) - \sum_{j\in\mathcal{A}(x)} \lambda_j c_j\right\| : \lambda_j \geq 0\right\}.$$

Also note that representation (4.4) of the projected gradient is unique even if the active constraints c_j with $j \in \mathcal{A}(x)$ are linearly dependent.

Theorem 4.1 characterizes the limiting behavior of many algorithms with respect to the face structure of Ω. For example, Burke and Moré [4] prove that the sequential quadratic programming algorithm and the gradient projection algorithm generate sequences that satisfy (4.2). Consequently, if the stationary point x^* is nondegenerate, the sequences generated by these algorithms eventually enter and remain in the relative interior of the face $F(x^*)$. This can be extremely useful knowledge in the design of algorithms for the solution of (1.1).

Note that Theorem 4.1 gives information on the identification of the active constraints. This is based on the observation of Burke and Moré [4] that

$$\operatorname{ri}\{F(x^*)\} = \{x \in \Omega : \langle c_j, x\rangle = \delta_j,\ j \in \mathcal{A}(x^*), \quad \langle c_j, x\rangle > \delta_j, \quad j \notin \mathcal{A}(x^*)\}.$$

Thus, Theorem 4.1 shows that if $\{x_k\}$ is a sequence in Ω that converges to a nondegenerate stationary point x^* of (1.1), then the sequence $\{P_{T(x_k)}\left[-\nabla f(x_k)\right]\}$ converges to zero if and only if there is a $k_0 > 0$ such that

$$\mathcal{A}(x_k) = \mathcal{A}(x^*), \qquad k \geq k_0.$$

The following example of Stephen Wright shows that this result fails in degenerate cases.

Example. Define $f : \mathbb{R}^2 \to \mathbb{R}$ by

$$f(\xi_1, \xi_2) = \tfrac{1}{2}\left(\xi_1^2 + \mu\xi_2^2\right)$$

for some $\mu > 1$, and define $\Omega \subset \mathbb{R}^2$ by

$$\Omega = \left\{(\xi_1, \xi_2) \in \mathbb{R}^2 : \xi_1 \geq \mu^2\xi_2,\ \xi_1 \geq -\xi_2\right\}.$$

Note that f is a strictly convex quadratic and that $x^* = 0$ is the global minimizer in Ω. A computation shows that the steepest descent iterates (with exact line searches) are generated by

$$\begin{pmatrix} \xi_1 \\ \xi_2 \end{pmatrix}_+ = \frac{\mu - 1}{\xi_1^2 + \mu^3\xi_2^2} \begin{pmatrix} \mu^2\xi_1\xi_2^2 \\ -\xi_1^2\xi_2 \end{pmatrix}.$$

This implies, in particular, that all the iterates are feasible and that

$$\frac{\xi_1^{(k+2)}}{\xi_2^{(k+2)}} = \frac{\xi_1^{(k)}}{\xi_2^{(k)}}.$$

Hence, every other iterate of the steepest descent method lies in the same ray from the origin. Thus, if the initial iterate satisfies either the constraint $\xi_1 = \mu^2\xi_2$ or the constraint $\xi_1 = -\xi_2$, the iterates of the steepest descent method oscillate between the two constraints that define Ω. Since these iterates are feasible, the steepest descent iterates coincide with the gradient projection iterates. This implies that in this example,

$$\mathcal{A}(x_{k+1}) \neq \mathcal{A}(x_k), \quad \mathcal{A}(x_k) \neq \mathcal{A}(x^*), \quad k \geq k_0.$$

Of course, in this case x^* is degenerate because $\nabla f(x^*) = 0$.

We now consider the case when x^* is degenerate and show that the nondegeneracy assumption in Theorem 4.1 can be dropped if we replace $\operatorname{ri}\{F(x^*)\}$ by $E[-\nabla f(x^*)]$.

THEOREM 4.2. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable on the polyhedral set Ω, and assume that $\{x_k\}$ is a sequence in Ω that converges to a stationary point x^* of (1.1). Then*

$$\lim_{k\to\infty} P_{T(x_k)}\left[-\nabla f(x_k)\right] = 0$$

if and only if there is a $k_0 > 0$ such that

$$x_k \in E[-\nabla f(x^*)], \qquad k \geq k_0.$$

Proof. The proof depends on the Moreau decomposition (see, for example, Lemma 2.2 of Zarantonello [28]) and Theorem 3.2. The Moreau decomposition of $-\nabla f$ shows that

$$-\nabla f(x_k) = -P_{T(x_k)}\left[-\nabla f(x_k)\right] + P_{N(x_k)}[-\nabla f(x_k)].$$

In particular,

$$-\nabla f(x_k) = -P_{T(x_k)}\left[-\nabla f(x_k)\right] + d_k,$$

where $d_k \in N(x_k)$. Hence, if the sequence $\{P_{T(x_k)}[-\nabla f(x_k)]\}$ converges to zero, then $\{d_k\}$ converges to $-\nabla f(x^*)$, and thus Theorem 3.2 shows that $x_k \in E[-\nabla f(x^*)]$ for all k sufficiently large.

Conversely, assume that $x_k \in E[-\nabla f(x^*)]$ for all k sufficiently large. Since x is in $E(d)$ if and only if $d \in N(x)$, we have that there is a $k_0 > 0$ such that $-\nabla f(x^*)$ belongs to $N(x_k)$ for all $k \geq k_0$. Since the Moreau decomposition implies that

$$\left\|P_{T(x_k)}[-\nabla f(x_k)]\right\| = \min\{\|\nabla f(x_k) + d\| : d \in N(x_k)\},$$

we obtain that

$$\left\|P_{T(x_k)}[-\nabla f(x_k)]\right\| \leq \|\nabla f(x_k) - \nabla f(x^*)\|.$$

This yields the desired result. □

In the remainder of this section we examine various consequences of Theorem 4.2. We first use Burke's result (Theorem 2.4) to show that Theorem 4.2 is an extension of Theorem 4.1 for polyhedral Ω.

THEOREM 4.3. *If Ω is a polyhedral set, then x^* is a nondegenerate stationary point for problem* (1.1) *if and only if $x^* \in \operatorname{ri}\{E[-\nabla f(x^*)]\}$.*

Proof. This result is a direct consequence of Theorem 2.4. □

We can show that Theorem 4.2 is an extension of Theorem 4.1 by noting that if x^* is a nondegenerate stationary point for problem (1.1), then Theorem 4.3 shows that x^* is in $\operatorname{ri}\{E[-\nabla f(x^*)]\}$, and thus any $x \in E[-\nabla f(x^*)]$ sufficiently close to x^* also belongs to $\operatorname{ri}\{E[-\nabla f(x^*)]\}$.

We now show that if the polyhedral set Ω has the specific representation (4.3), then the face $E[-\nabla f(x^*)]$ can be expressed in terms of the positive Lagrange multipliers at x^*.

THEOREM 4.4. *If Ω is the polyhedral set in $\mathbb{R}^n$ defined by* (4.3) *and*

$$\nabla f(x^*) = \sum_{j \in \mathcal{A}(x^*)} \lambda_j^* c_j, \qquad \lambda_j^* \geq 0, \tag{4.5}$$

for some stationary point x^* of problem (1.1), *then*

$$E[-\nabla f(x^*)] = \left\{x \in \Omega : \langle c_j, x\rangle = \delta_j,\ \lambda_j^* > 0\right\}.$$

Proof. Note that

$$\langle \nabla f(x^*), x\rangle = \sum_{j \in \mathcal{A}(x^*)} \lambda_j^* \langle c_j, x\rangle \geq \sum_{j \in \mathcal{A}(x^*)} \lambda_j^* \delta_j$$

for any $x \in \Omega$. Thus, we have that $\langle \nabla f(x^*), x\rangle$ is minimized only when $\langle c_j, x\rangle = \delta_j$ for $\lambda_j^* > 0$. □

If the active constraint normals are linearly dependent then there is an infinite set of Lagrange multipliers that satisfy (4.5). Nevertheless, Theorem 4.4 shows that the set

$$\left\{x \in \Omega : \langle c_j, x\rangle = \delta_j,\ \lambda_j^* > 0\right\}$$

is independent of the choice of Lagrange multipliers.

THEOREM 4.5. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable on the polyhedral set Ω defined by* (4.3). *Assume that x^* is a stationary point of* (1.1) *and that the*

Lagrange multipliers λ_i^* satisfy (4.5). *If $\{x_k\}$ is a sequence in Ω that converges to x^*, then*

$$\lim_{k\to\infty} P_{T(x_k)}\left[-\nabla f(x_k)\right] = 0 \tag{4.6}$$

if and only if there is a $k_0 > 0$ such that

$$\{i \in \mathcal{A}(x^*) : \lambda_i^* > 0\} \subset \mathcal{A}(x_k), \qquad k \geq k_0. \tag{4.7}$$

Proof. Theorem 4.4 shows that

$$x \in E[-\nabla f(x^*)] \iff \{i \in \mathcal{A}(x^*) : \lambda_i^* > 0\} \subset \mathcal{A}(x), \quad x \in \Omega.$$

Thus, the result is a direct consequence of Theorem 4.2. □

This result has immediate applications to several algorithms. For example, Calamai and Moré [6] show that the gradient projection method generates sequences that satisfy (4.6). Hence, Theorem 4.5 implies that if the sequence $\{x_k\}$ generated by the gradient projection method converges to x^*, then (4.7) holds.

Closely related convergence results for the gradient projection method are due to Bertsekas [1], [2], Gafni and Bertsekas [12], [13], and Dunn [9]. However, in these papers it is not shown that (4.6) holds; instead, it is shown that any limit point of the sequence $\{x_k\}$ is stationary.

Theorem 4.5 also has application to the class of trust region methods for bound constrained optimization problems proposed by Conn, Gould, and Toint [7], [8]. For this algorithm, Lescrenier [17] proved that (4.7) holds. Hence, Theorem 4.5 implies that (4.6) holds. This result is of interest because the measure

$$\nu(x; f) = \left\|P_{T(x)}\left[-\nabla f(x)\right]\right\| \tag{4.8}$$

is a natural criterion for terminating the algorithm. This is clear from expression (4.4) for the projected gradient. Moreover, note that $\nu(x;\cdot)$ is scale invariant in the sense that $\nu(x;\alpha f) = \alpha\nu(x, f)$ for any $\alpha > 0$. Other criteria, for example,

$$\nu(x; f) = \|P[x - \nabla f(x)] - x\|$$

do not have this important property.

Finally, Theorem 4.5 has implications to the class of trust region methods for general linearly constrained methods analyzed by Burke, Moré, and Toraldo [5].

5. Strictly binding constraints. We have stressed the geometric viewpoint of Theorem 4.2 because this viewpoint leads to results that are independent of the representation of Ω. On the other hand, the viewpoint of Theorem 4.5 is needed because it is closely related to computational issues. In this section we show that it is possible to define a set of strictly binding constraints $\mathcal{B}_s^*$ at x^*, independent of the choice of Lagrange multipliers. This leads, in particular, to a version of Theorem 4.5 in terms of $\mathcal{B}_s^*$.

We will show that the set of strictly binding constraints $\mathcal{B}_s^*$ can be defined in terms of the active set of the exposed face $E[-\nabla f(x^*)]$.

DEFINITION. *If Ω is a polyhedral set and F is a face of Ω, then*

$$\mathcal{A}(F) = \mathcal{A}(x), \qquad x \in \operatorname{ri}\{F\}$$

is the active set of F.

We justify this definition by showing that the active set $\mathcal{A}(x)$ is independent of x in $\operatorname{ri}\{F\}$. Assume that Ω is defined by (4.3), and choose any $x \in \operatorname{ri}\{F\}$. If $y \in F$, then

$$x_\lambda = x + \lambda(y - x) \in F$$

for some $\lambda < 0$. Hence, if $i \in \mathcal{A}(x)$, then

$$\delta_i \leq \langle c_i, x_\lambda \rangle = (1 - \lambda)\langle c_i, x \rangle + \lambda \langle c_i, y \rangle = (1 - \lambda)\delta_i + \lambda \langle c_i, y \rangle.$$

Since $\lambda < 0$, this implies that $\langle c_i, y \rangle \leq \delta_i$. Thus, $i \in \mathcal{A}(y)$. We have shown that $\mathcal{A}(x) \subset \mathcal{A}(y)$ if $x \in \operatorname{ri}\{F\}$ and $y \in F$. Hence, $\mathcal{A}(x) = \mathcal{A}(y)$ if x and y belong to $\operatorname{ri}\{F\}$.

This definition of $\mathcal{A}(F)$ is direct and intuitive. We now establish an alternative characterization of $\mathcal{A}(F)$. For the proof of this result, we need to know that

$$-d \in \operatorname{ri}\{N(x)\} \iff d = \sum_{i \in \mathcal{A}(x)} \lambda_i c_i, \quad \lambda_i > 0. \tag{5.1}$$

This characterization of $\operatorname{ri}\{N(x)\}$ is due to Burke and Moré [4, Lemma 3.2].

LEMMA 5.1. *If* Ω *is a polyhedral set, then*

$$F = \{x \in \Omega : \langle c_i, x \rangle = \delta_i, \ i \in \mathcal{A}(F)\}$$

for any face F *of* Ω.

Proof. Every face of a polyhedron is exposed (see, for example, Thm. 2.4.12 of Stoer and Witzgall [25]), and thus $F = E(-d)$ for some $d \in \mathbb{R}^n$. Hence, Theorem 2.4 implies that $-d \in \operatorname{ri}\{N(F)\}$, and thus $d \in \operatorname{ri}\{N(x)\}$ for any $x \in \operatorname{ri}\{F\}$. We now appeal to the characterization (5.1) and conclude that there are $\lambda_i > 0$ such that

$$d = \sum_{i \in \mathcal{A}(x)} \lambda_i c_i, \qquad \lambda_i > 0.$$

A computation now shows that

$$E[-d] = \{x \in \Omega : \langle c_i, x \rangle = \delta_i, \ i \in \mathcal{A}(x)\}.$$

Since $\mathcal{A}(x) = \mathcal{A}(F)$ and $F = E[-d]$, this establishes the result. □

This result characterizes $\mathcal{A}(F)$ as the set of active constraints that are essential for F. In view of this result, it is natural to investigate the properties of the active set of the exposed face $E[-\nabla f(x^*)]$.

DEFINITION. *If* Ω *is a polyhedral set and* x^* *is a stationary point of* (1.1), *then*

$$\mathcal{B}_s^* = \mathcal{A}\big(E[-\nabla f(x^*)]\big)$$

is the set of strictly binding constraints at x^*.

This definition requires justification because strictly binding constraints are usually associated with positive Lagrange multipliers; the following result provides this justification.

THEOREM 5.2. *Let* $f : \mathbb{R}^n \to \mathbb{R}$ *be continuously differentiable on the polyhedral set* Ω *defined by* (4.3). *If* x^* *is a stationary point of* (1.1) *then*

$$\nabla f(x^*) = \sum_{j \in \mathcal{B}_s^*} \lambda_j^* c_j, \qquad \lambda_j^* > 0, \tag{5.2}$$

for some set of positive Lagrange multipliers λ_i^*.

Proof. Theorem 2.4 implies that $-\nabla f(x^*) \in \operatorname{ri}\{N(F)\}$ for $F = E[-\nabla f(x^*)]$. Hence, $-\nabla f(x^*) \in \operatorname{ri}\{N(x)\}$ for any $x \in \operatorname{ri}\{F\}$. Since $\mathcal{A}(x) = \mathcal{B}_s^*$ for $x \in \operatorname{ri}\{F\}$, the characterization (5.1) yields the result. □

Theorem 5.2 justifies the definition of $\mathcal{B}_s^*$ as the set of strictly binding constraints. This result is interesting because it does not make any nondegeneracy or linear independence assumptions. Note that if x^* is nondegenerate then Theorem 4.3 shows that $x^* \in \operatorname{ri}\{E[-\nabla f(x^*)]\}$, and thus $\mathcal{B}_s^* = \mathcal{A}(x^*)$. Hence, for nondegenerate x^*, Theorem 5.2 is a direct consequence of (5.1) and the definition of nondegeneracy.

THEOREM 5.3. *Let* $f : \mathbf{R}^n \to \mathbf{R}$ *be continuously differentiable on the polyhedral set* Ω *defined by* (4.3). *If* x^* *is a stationary point of* (1.1) *and* $\{\lambda_i^*\}$ *is any set of Lagrange multipliers, then the following three statements are equivalent.*

1. $x \in E[-\nabla f(x^*)]$.
2. $\{i \in \mathcal{A}(x^*) : \lambda_i^* > 0\} \subset \mathcal{A}(x)$ *for* $x \in \Omega$.
3. $\mathcal{B}_s^* \subset \mathcal{A}(x)$ *for* $x \in \Omega$.

Proof. Theorem 4.4 shows that the first two statements are equivalent. The equivalence of the first and third statements is obtained by noting that Lemma 5.1 implies that

$$E[-\nabla f(x^*)] = \{x \in \Omega : \langle c_i, x\rangle = \delta_i,\ i \in \mathcal{B}_s^*\}. \qquad \square$$

This result shows that Theorem 4.5 can be phrased in terms of $\mathcal{B}_s^*$. Another interesting consequence of Theorem 5.3 is obtained by choosing any $x \in \operatorname{ri}\{E[-\nabla f(x^*)]\}$ in the second statement in Theorem 5.3. This shows that

$$\{i \in \mathcal{A}(x^*) : \lambda_i^* > 0\} \subset \mathcal{B}_s^*$$

for any set of Lagrange multipliers $\{\lambda_i^*\}$. Hence, the indices in $\mathcal{B}_s^*$ identify a maximal set of positive Lagrange multipliers.

6. Second-order sufficiency conditions. We have shown that optimization algorithms tend to generate iterates such that

$$x_k \in E[-\nabla f(x^*)]$$

for all k sufficiently large. Another important component in the analysis of an optimization algorithm is to show that the iterates converge under suitable conditions. This type of result usually requires the assumption that the iterates $\{x_k\}$ have an isolated limit point x^*. This assumption is satisfied, for example, if x^* is an *isolated* stationary point; that is, there is a neighborhood $S(x^*)$ such that x^* is the only stationary point in $S(x^*) \cap \Omega$. In this section we show that if the Hessian is positive definite in the cone generated by $E[-\nabla f(x^*)] - x^*$, then the stationary point x^* is isolated.

The assumption that x^* is an isolated stationary point can be guaranteed by imposing second-order conditions on f. The following result uses a version of the second-order sufficiency conditions that is appropriate for problem (1.1) with a general convex set Ω.

THEOREM 6.1. *Let* $f : \mathbf{R}^n \to \mathbf{R}$ *be continuously differentiable on a closed convex set* Ω *and twice differentiable at a point* x^* *in* Ω. *If* x^* *is a stationary point of problem* (1.1) *and*

$$\langle \nabla f(x^*), w\rangle = 0, \quad w \in T(x^*), \quad w \neq 0 \quad \Longrightarrow \quad \langle w, \nabla^2 f(x^*) w\rangle > 0, \tag{6.1}$$

then x^* is an isolated stationary point of f.

Theorem 6.1 is a special case of a result of Robinson [21, Thm. 2.4]. The proof of Theorem 6.1 is not difficult in our setting. Note that if x^* is not an isolated stationary point, then there is a sequence $\{x_k\}$ of stationary points converging to x^*. In particular,

$$\langle \nabla f(x^*), x_k - x^* \rangle \geq 0, \qquad \langle \nabla f(x_k), x^* - x_k \rangle \geq 0, \tag{6.2}$$

This implies that

$$\langle \nabla f(x_k) - \nabla f(x^*), x_k - x^* \rangle \leq 0. \tag{6.3}$$

Hence, if w is a limit point of the sequence $\{w_k\}$ defined by

$$w_k = \frac{x_k - x^*}{\|x_k - x^*\|},$$

then (6.2) shows that $\langle \nabla f(x^*), w \rangle = 0$, while (6.3) shows that $\langle w, \nabla^2 f(x^*) w \rangle \leq 0$. Moreover, since $w_k \in T(x^*)$ and $\|w_k\| = 1$, we also have that $w \in T(x^*)$ and that $\|w\| = 1$. This contradicts (6.1).

For a polyhedral Ω, condition (6.1) coincides with the standard second-order sufficiency conditions. This observation follows by noting that if Ω is the set defined by (4.3) and

$$\nabla f(x^*) = \sum_{j \in \mathcal{A}(x^*)} \lambda_j^* c_j, \qquad \lambda_j^* \geq 0,$$

then

$$\{w \in \mathbb{R}^n : \langle \nabla f(x^*), w \rangle = 0, \quad w \in T(x^*)\} = \\ \{w \in \mathbb{R}^n : \langle c_j, w \rangle \geq 0, \ j \in \mathcal{A}(x^*), \quad \langle c_j, w \rangle = 0, \ \lambda_j^* > 0\}.$$

For a general convex Ω, condition (6.1) has an advantage over the standard second-order sufficiency conditions because it is independent of the representation of Ω. On the other hand, an example of Burke, Moré, and Toraldo [5] shows that condition (6.1) does not take into account the curvature of Ω, and thus differs from the standard second-order sufficiency conditions.

If x^* is a nondegenerate stationary point, then (6.1) can be expressed in terms of $N(x^*)^\perp$, where for any set S the orthogonal complement $S^\perp$ of S is the subspace of vectors v such that $\langle v, w \rangle = 0$ for all $w \in S$. Indeed, Burke, Moré, and Toraldo [5] prove that if x^* is a nondegenerate stationary point, then

$$N(x^*)^\perp = \{w \in T(x^*) : \langle \nabla f(x^*), w \rangle = 0\}. \tag{6.4}$$

These results are familiar when Ω is the polyhedral set defined by (4.3) because

$$N(x^*)^\perp = \{v \in \mathbb{R}^n : \langle c_j, v \rangle = 0, \ j \in \mathcal{A}(x^*)\}.$$

We now show that the second-order sufficiency condition can be expressed in terms of the exposed face $E[-\nabla f(x^*)]$. In this result, cone $\{S\}$ is the cone spanned by the set S, that is, the set of vectors αw for some $\alpha \geq 0$ and $w \in S$.

THEOREM 6.2. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable on a polyhedral Ω and twice differentiable at a point x^* in Ω. If x^* is a stationary point of problem (1.1), then*

$$\{w \in T(x^*) : \langle \nabla f(x^*), w \rangle = 0\} = \operatorname{cone}\{E[-\nabla f(x^*)] - x^*\}. \tag{6.5}$$

Moreover, if

$$w \in \operatorname{cone}\{E[-\nabla f(x^*)] - x^*\}, \quad w \neq 0 \implies \langle w, \nabla^2 f(x^*) w \rangle > 0, \tag{6.6}$$

then x^* is an isolated stationary point of f.

Proof. The result follows from Theorem 6.1 if we establish (6.5). We first show that

$$\operatorname{cone}\{E[-\nabla f(x^*)] - x^*\} \subset \{w \in T(x^*) : \langle \nabla f(x^*), w \rangle = 0\}.$$

If $w \in \operatorname{cone}\{E[-\nabla f(x^*)] - x^*\}$, then $w = \alpha(x - x^*)$ for some $x \in E[-\nabla f(x^*)]$. Thus, since $\langle \nabla f(x^*), \cdot \rangle$ is constant on $E[-\nabla f(x^*)]$, we obtain that $\langle \nabla f(x^*), w \rangle = 0$. Moreover, since $x \in \Omega$, it is clear that $w \in T(x^*)$. For the reverse inclusion assume that $\langle \nabla f(x^*), w \rangle = 0$ for some $w \in T(x^*)$. Since $w \in T(x^*)$ and Ω is polyhedral, $x^* + \alpha w \in \Omega$ for all $\alpha > 0$ sufficiently small. Moreover, since $\langle \nabla f(x^*), w \rangle = 0$, it follows that $x^* + \alpha w \in E[-\nabla f(x^*)]$. Thus, we have $w \in \operatorname{cone}\{E[-\nabla f(x^*)] - x^*\}$ as desired. □

We have already noted that if x^* is nondegenerate, then (6.4) holds. This implies, in particular, that

$$\{w \in T(x^*) : \langle \nabla f(x^*), w \rangle = 0\}$$

is a subspace. In view of (6.5), we have shown that if x^* is nondegenerate then the second-order condition (6.1) reduces to the assumption that $\nabla^2 f(x^*)$ is positive definite on the subspace $\operatorname{aff}\{E[-\nabla f(x^*)] - x^*\}$. In the context of algorithms, the fact that (6.5) is a subspace is extremely helpful. However, the same effect can be obtained without assuming nondegeneracy if one is willing to strengthen the second-order sufficiency condition.

DEFINITION. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be twice differentiable at $x^* \in \Omega$. The strong second-order sufficiency condition for problem (1.1) is satisfied at a stationary point x^* if $\nabla^2 f(x^*)$ is positive definite on the subspace* $\operatorname{aff}\{E[-\nabla f(x^*)] - x^*\}$; *that is,*

$$w \in \operatorname{aff}\{E[-\nabla f(x^*)] - x^*\}, \quad w \neq 0 \implies \langle w, \nabla^2 f(x^*) w \rangle > 0. \tag{6.7}$$

If the strong second-order sufficiency condition (6.7) holds, then the standard second-order sufficiency condition (6.1) must also hold because

$$\operatorname{cone}\{E[-\nabla f(x^*)] - x^*\} \subset \operatorname{aff}\{E[-\nabla f(x^*)] - x^*\}.$$

The following example shows that the converse can fail even if f is quadratic.

Example. Let $f : \mathbb{R}^2 \to \mathbb{R}$ be given by

$$f(\xi_1, \xi_2) = \xi_1^2 + 4\xi_1\xi_2 + \xi_2^2,$$

and let $\Omega \subset \mathbb{R}^2$ be given by

$$\Omega = \{(\xi_1, \xi_2) \in \mathbb{R}^2 : \xi_1 \geq 0,\ \xi_2 \geq 0\}.$$

For this problem $x^* = (0,0)$ is the global minimum of f over Ω. Note that $\nabla f(x^*) = 0$, and thus $E[-\nabla f(x^*)] = \Omega$. Hence,

$$\operatorname{cone}\{E[-\nabla f(x^*)] - x^*\} = \mathbb{R}^2_+, \qquad \operatorname{aff}\{E[-\nabla f(x^*)] - x^*\} = \mathbb{R}^2.$$

A computation shows that the standard second-order sufficiency condition (6.1) holds at x^*. However, it is clear that the strong second-order sufficiency condition (6.7) is not satisfied at x^* because $\nabla^2 f(x^*)$ is not positive definite.

Other authors have used the term strong second-order sufficiency condition to mean that

$$\langle c_j, w\rangle = 0,\ \lambda_j^* > 0, \qquad w \neq 0 \implies \langle w, \nabla^2 f(x^*) w\rangle > 0. \tag{6.8}$$

For example, Gay [14], Lescrenier [17], and Robinson [20] use this condition in their analysis of the convergence behavior of algorithms. A disadvantage of condition (6.8) is that it depends on the representation of Ω and the choice of multipliers. The following example shows that condition (6.8) is stronger than (6.7) even if f is a quadratic.

Example. Let $f : \mathbb{R}^3 \to \mathbb{R}$ be given by

$$f(\xi_1, \xi_2, \xi_3) = \xi_1^2 + \xi_3^2 - \xi_2^2 + \xi_3,$$

and let $\Omega = \{x \in \mathbb{R}^3 : \langle c_i, x\rangle \geq 0,\ 1 \leq i \leq 4\}$ be the cone with vertex at the origin, where

$$c_1 = e_1, \quad c_2 = e_2, \quad c_3 = e_3 - e_1, \quad c_4 = e_3 - e_2.$$

For this problem $x^* = (0,0,0)$ is the global minimum of f over Ω. A computation shows that $E[-\nabla f(x^*)] = \{x^*\}$, and thus the strong second-order condition (6.7) holds. However, condition (6.8) does not hold if we choose $\lambda_2^* = \lambda_4^* = 0$.

This example shows that condition (6.8) is stronger than (6.7) for some set of multipliers. On the other hand, the following result shows that there is a set of multipliers such that condition (6.8) is equivalent to (6.7).

THEOREM 6.3. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable on the polyhedral set Ω defined by* (4.3). *If x^* is a stationary point of* (1.1), *then*

$$\operatorname{aff}\{E[-\nabla f(x^*)] - x^*\} = \{v \in \mathbb{R}^n : \langle c_j, v\rangle = 0,\ j \in \mathcal{B}_s^*\},$$

where $\mathcal{B}_s^*$ is the set of strictly binding constraints defined in §5.

Proof. If $v \in E[-\nabla f(x^*)] - x^*$, then $\langle \nabla f(x^*), v\rangle = 0$, because $\langle \nabla f(x^*), \cdot\rangle$ is constant on $E[-\nabla f(x^*)]$. Moreover, since v is a feasible direction, $\langle c_j, v\rangle \geq 0$ for $j \in \mathcal{A}(x^*)$. Now recall that Theorem 5.2 guarantees that (5.2) holds. Hence,

$$0 = \langle \nabla f(x^*), v\rangle = \sum_{j \in \mathcal{B}_s^*} \lambda_j^* \langle c_j, v\rangle \geq 0.$$

This implies that $\langle c_j, v\rangle = 0$ when $j \in \mathcal{B}_s^*$. Thus, we have shown that

$$E[-\nabla f(x^*)] - x^* \subset \{v \in \mathbb{R}^n : \langle c_j, v\rangle = 0,\ j \in \mathcal{B}_s^*\}.$$

Hence,

$$\operatorname{aff}\{E[-\nabla f(x^*)] - x^*\} \subset \{v \in \mathbb{R}^n : \langle c_j, v\rangle = 0,\ j \in \mathcal{B}_s^*\}. \tag{6.9}$$

For the reverse inclusion choose any $x_0 \in \operatorname{ri}\{E[-\nabla f(x^*)]\}$. If $\langle c_j, v\rangle = 0$ for $j \in \mathcal{B}_s^*$, then v is a feasible direction because $\mathcal{B}_s^* = \mathcal{A}(x_0)$. Hence $x_0 + \alpha v \in \Omega$ for all $\alpha > 0$ sufficiently small. Moreover, (5.2) implies that

$$\langle \nabla f(x^*), x + \alpha v\rangle = \langle \nabla f(x^*), x\rangle.$$

This shows that $x_0 + \alpha v \in E[-\nabla f(x^*)]$. We have established that

$$\{v \in \mathbb{R}^n : \langle c_j, v\rangle = 0,\ j \in \mathcal{B}_s^*\} \subset E[-\nabla f(x^*)] - x_0.$$

Hence,

$$\dim\{v \in \mathbb{R}^n : \langle c_j, v\rangle = 0,\ j \in \mathcal{B}_s^*\} \le \dim\{E[-\nabla f(x^*)]\}.$$

Since we have already established (6.9), this completes the proof. □

7. Algorithms. Theorem 4.5 is an important tool for the convergence analysis of optimization algorithms. In this section we examine the use of Theorem 4.5 in the GPCG algorithm of Moré and Toraldo [19]. In particular, we show that the convergence condition of the GPCG algorithm is satisfied in a finite number of iterations even when the solution is degenerate.

The GPCG algorithm uses a combination of the gradient projection algorithm and the conjugate gradient algorithm to solve large-scale problems of the form

$$\min\{q(x) : l \le x \le u\}. \tag{7.1}$$

The GPCG algorithm uses the conjugate gradient method to explore the active set

$$\mathcal{A}(x) = \{i : x_i \in \{l_i, u_i\}\} \tag{7.2}$$

defined by the current iterate. Once this exploration is completed, the gradient projection method is used to choose a new active set. The convergence properties of the GPCG algorithm are summarized in the following result of Moré and Toraldo [19].

THEOREM 7.1. *Let $q : \mathbb{R}^n \to \mathbb{R}$ be a strictly convex quadratic. If $\{x_k\}$ is the sequence generated by the* GPCG *algorithm for problem* (7.1), *then either $\{x_k\}$ terminates at the solution x^* in a finite number of iterations, or $\{x_k\}$ converges to the solution x^* of problem* (7.1). *If the solution x^* of problem* (7.1) *satisfies the nondegeneracy condition*

$$\partial_i q(x^*) \ne 0, \qquad i \in \mathcal{A}(x^*),$$

then the GPCG *algorithm terminates at the solution x^* in a finite number of iterations.*

This convergence result is not entirely satisfactory when the solution x^* is degenerate, because it does not show that the convergence test is satisfied in a finite number of iterations. Given a starting point x_0 in the feasible region $\Omega = \{x \in \mathbb{R}^n : l \le x \le u\}$ and a tolerance $\tau > 0$, the convergence test of the GPCG algorithm requires that

$$\|\nabla_\Omega q(x_k)\| \le \tau \|\nabla q(x_0)\|, \tag{7.3}$$

where $\nabla_\Omega q$ is defined by

$$[\nabla_\Omega q(x)]_i = \begin{cases} \partial_i q(x) & \text{if } x_i \in (l_i, u_i), \\ \min\{\partial_i q(x), 0\} & \text{if } x_i = l_i, \\ \max\{\partial_i q(x), 0\} & \text{if } x_i = u_i. \end{cases}$$

This convergence test is closely related to (4.6) because

$$\nabla_\Omega q(x) = -P_{T(x)}\,[-\nabla f(x)].$$

In particular, if (4.6) holds, then the convergence test (7.3) is satisfied after a finite number of iterations. In the remainder of this section we show that (4.6) holds.

Given the current iterate x_k, the GPCG algorithm explores the active set defined by the current iterate by computing an approximate minimizer of the subproblem

$$\min\{q(x_k + d) : d_i = 0,\ i \in \mathcal{A}(x_k)\}. \tag{7.4}$$

Given an approximate minimizer d_k of subproblem (7.4), the GPCG algorithm uses a projected search to choose a search parameter α_k such that $q(x_{k+1}) < q(x_k)$, where

$$x_{k+1} = P(x_k + \alpha_k d_k)$$

and P is the projection into the feasible region Ω. Details on the projected search can be found in the paper of Moré and Toraldo [19]. For this paper it is necessary to note only that since the ith coordinate of d_k is zero when $i \in \mathcal{A}(x_k)$, we obtain that

$$\mathcal{A}(x_k) \subset \mathcal{A}(x_{k+1}) \tag{7.5}$$

holds whenever the GPCG algorithm explores the active set defined by the current iterate.

The approximate minimizer d_k of subproblem (7.4) is obtained by first noting that if $i_1, \ldots, i_{m_k}$ are the indices of the *free* variables, that is, those variables with indices outside of $\mathcal{A}(x_k)$, then subproblem (7.4) is equivalent to the unconstrained subproblem

$$\min\{q_k(w) : w \in \mathbb{R}^{m_k}\}, \tag{7.6}$$

where $q_k : \mathbb{R}^{m_k} \to \mathbb{R}$ is defined by $q_k(w) \equiv q(x_k + Z_k w)$, and Z_k is the matrix in $\mathbb{R}^{n \times m_k}$ whose jth column is the i_jth column of the identity matrix in $\mathbb{R}^{n \times n}$. Given the starting point $w_0 = 0$ in $\mathbb{R}^{m_k}$, the conjugate gradient algorithm generates a sequence of iterates $\{w_{j_k}\}$. The approximate solution of subproblem (7.4) is then $d_k = Z_k w_{j_k}$ for some $j_k \geq 0$.

If the iterate x_{k+1} generated by the conjugate gradient method appears to have identified the active set defined by the solution, then the GPCG algorithm explores this active set further. The decision to continue the conjugate gradient method is based on the observation that if $\mathcal{A}(x) = \mathcal{A}(x^*)$, then the *binding set*

$$\mathcal{B}(x) = \{i : x_i = l_i \text{ and } \partial_i q(x) \geq 0,\ \text{or } x_i = u_i \text{ and } \partial_i q(x) \leq 0\}$$

agrees with the active set $\mathcal{A}(x)$. Thus, if the conjugate gradient method produces an iterate x_{k+1} such that $\mathcal{B}(x_{k+1}) = \mathcal{A}(x_{k+1})$, then the GPCG algorithm continues to use the conjugate gradient method to explore this active set. The finite termination properties of the conjugate gradient algorithm show that at most m_k iterations are needed before the conjugate gradient algorithm finds a solution of subproblem (7.4); at this point there would be no need to explore this active set further.

Once the conjugate gradient algorithm has explored an active set, the GPCG algorithm uses the gradient projection method

$$y_{k+1} = P\,[y_k - \alpha_k \nabla q(y_k)]$$

with $y_0 = x_k$ to select a new active set. The GPCG algorithm sets $x_{k+1} = y_{j_k}$ for some $j_k \geq 0$. Additional details on the implementation of the gradient projection method can be found in the paper of Moré and Toraldo [19].

We now use Theorem 4.5 to prove that (4.6) holds for the iterates generated by the GPCG algorithm. The proof consists of showing that condition (4.7) holds. For the bound constrained problem (7.1), condition (4.7) is just that

$$\mathcal{B}_s^* \equiv \{i \in \mathcal{A}(x^*) : \partial_i q(x^*) \neq 0\} \subset \mathcal{A}(x_k), \qquad k \geq k_0,$$

where $\mathcal{B}_s^*$ is the set of strictly binding constraints at x^* defined in §5.

THEOREM 7.2. *Let $q : \mathbb{R}^n \to \mathbb{R}$ be a strictly convex quadratic. If $\{x_k\}$ is the sequence generated by the* GPCG *algorithm for problem* (7.1), *then either $\{x_k\}$ terminates at the solution x^* in a finite number of iterations, or there is an index k_0 such that the convergence test* (7.3) *is satisfied for all $k \geq k_0$.*

Proof. Theorem 7.1 shows that either $\{x_k\}$ terminates at the solution x^* in a finite number of iterations or that $\{x_k\}$ converges to x^*. The result follows if $\{x_k\}$ terminates at the solution x^*, so we assume that $\{x_k\}$ converges to x^*. Let $\mathcal{K}_{GP}$ be the set of indices k such that x_{k+1} is generated by the gradient projection method. Theorem 5.2 of Calamai and Moré [6] shows that

$$\lim_{k \in \mathcal{K}_{GP}, k \to \infty} P_{T(x_{k+1})}\left[-\nabla q(x_{k+1})\right] = 0.$$

Hence, Theorem 4.6 shows that

$$\mathcal{B}_s^* \subset \mathcal{A}(x_{k+1})$$

if $k \in \mathcal{K}_{GP}$ and k is sufficiently large. Moreover, since (7.5) holds whenever the conjugate gradient method is used to explore the current active set, we immediately obtain that

$$\mathcal{B}_s^* \subset \mathcal{A}(x_k)$$

holds for all sufficiently large indices k. Hence, (4.7) holds, and thus Theorem 4.5 shows that (4.6) holds as desired. □

Acknowledgments. We thank Stephen Wright for his interesting example in §4 of a degenerate optimization problem where the active set does not settle down, and Steve Robinson for pointing out that Theorem 3.2 follows directly from [22, Lemma 3.5]. We are also grateful to Gail Pieper for her gentle comments on the manuscript.

REFERENCES

[1] D. P. BERTSEKAS, *On the Goldstein–Levitin–Polyak gradient projection method*, IEEE Trans. Automat. Control, 21 (1976), pp. 174–184.

[2] ———, *Projected Newton methods for optimization problems with simple constraints*, SIAM J. Control Optim., 20 (1982), pp. 221–246.

[3] J. V. BURKE, *On the identification of active constraints* II*: The nonconvex case*, SIAM J. Numer. Anal., 27 (1990), pp. 1081–1102.

[4] J. V. BURKE AND J. J. MORÉ, *On the identification of active constraints*, SIAM J. Numer. Anal., 25 (1988), pp. 1197–1211.

[5] J. V. BURKE, J. J. MORÉ, AND G. TORALDO, *Convergence properties of trust region methods for linear and convex constraints*, Math. Programming, 47 (1990), pp. 305–336.

[6] P. H. CALAMAI AND J. J. MORÉ, *Projected gradient methods for linearly constrained problems*, Math. Programming, 39 (1987), pp. 93–116.

[7] A. R. CONN, N. I. M. GOULD, AND P. L. TOINT, *Global convergence of a class of trust region algorithms for optimization problems with simple bounds*, SIAM J. Numer. Anal., 25 (1988), pp. 433–460.

[8] ———, *Testing a class of methods for solving minimization problems with simple bounds on the variables*, Math. Comp., 50 (1988), pp. 399–430.

[9] J. C. DUNN, *Global and asymptotic convergence rate estimates for a class of projected gradient processes*, SIAM J. Control Optim., 19 (1981), pp. 368–400.

[10] ———, *On the convergence of projected gradient processes to singular critical points*, J. Optim. Theory Appl., 55 (1987), pp. 203–216.

[11] ———, *A projected Newton method for minimization problems with nonlinear inequality constraints*, Numer. Math., 53 (1988), pp. 377–410.

[12] E. M. GAFNI AND D. P. BERTSEKAS, *Convergence of a gradient projection method*, Report LIDS-P-1201, Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA, 1982.

[13] ———, *Two-metric projection methods for constrained optimization*, SIAM J. Control Optim., 22 (1984), pp. 936–964.

[14] D. M. GAY, *A trust region approach to linearly constrained optimization*, in Numerical Analysis, D. F. Griffiths, ed., Lecture Notes in Mathematics 1066, Springer-Verlag, New York, 1984, pp. 72–105.

[15] J. L. GOFFIN, *The relaxation method for solving systems of linear inequalities*, Math. Oper. Res., 5 (1980), pp. 388–414.

[16] C. T. KELLEY AND E. W. SACHS, *Mesh independence of the gradient projection method for optimal control problems*, SIAM J. Control Optim., 30 (1992), pp. 477–493.

[17] M. LESCRENIER, *Convergence of trust region algorithms for optimization with bounds when strict complementarity does not hold*, SIAM J. Numer. Anal., 28 (1991), pp. 476–495.

[18] J. J. MORÉ, *Trust regions and projected gradients*, in Systems Modelling and Optimization, M. Iri and K. Yajima, eds., Lecture Notes in Control and Information Sciences 113, Springer-Verlag, New York, 1988, pp. 1–13.

[19] J. J. MORÉ AND G. TORALDO, *Algorithms for bound constrained quadratic programming problems*, Numer. Math., 55 (1989), pp. 377–400.

[20] S. M. ROBINSON, *Strongly regular generalized equations*, Math. Oper. Res., 5 (1980), pp. 43–62.

[21] ———, *Generalized equations and their solutions, part* II*: Applications to nonlinear programming*, Math. Programming Stud., 19 (1982), pp. 200–221.

[22] ———, *Local structure of feasible sets in nonlinear programming, part* III*: Nondegeneracy*, Math. Programming Stud., 22 (1984), pp. 217–230.

[23] ———, *Normal maps induced by linear transformations*, Math. Oper. Res., 17 (1992), pp. 691–714.

[24] R. T. ROCKAFELLAR, *Convex Analysis*, Princeton University Press, Princeton, NJ, 1970.

[25] J. STOER AND C. WITZGALL, *Convexity and Optimization in Finite Dimensions* I, Springer-Verlag, New York, 1970.

[26] S. J. WRIGHT, *Implementing proximal point methods for linear programming*, J. Optim. Theory Appl., 65 (1990), pp. 531–554.

[27] ———, *Identifiable surfaces in constrained optimization*, SIAM J. Control Optim., 31 (1993), pp. 1063–1079.

[28] E. H. ZARANTONELLO, *Projections on convex sets in Hilbert space and spectral theory*, in Contributions to Nonlinear Functional Analysis, E. H. Zarantonello, ed., Academic Press, New York, 1971, pp. 237–424.

SIAM J. OPTIMIZATION
Vol. 4, No. 3, pp. 596–612, August 1994
© 1994 Society for Industrial and Applied Mathematics
009

POTENTIAL REDUCTION POLYNOMIAL TIME METHOD FOR TRUSS TOPOLOGY DESIGN*

AHARON BEN-TAL† AND ARKADII NEMIROVSKII†

Abstract. The multiload truss topology design problem is modeled as a minimization of the maximum (with respect to k loading scenarios) compliance subject to equilibium constraints and restrictions on the bar volumes. The problem may involve a very large number of potential bars so as to allow a rich variety of topologies. The original formulation is in terms of two sets of variables: m bar volumes and n nodal displacements. Several equivalent convex reformulations of this problem are presented. These convex problems, although highly nonlinear, possess nice analytical structure, and therefore can be solved by an interior point potential reduction method associated with appropriate logarithmic barrier for the feasible domain of the problem. For this method, to improve the accuracy of the current approximate solution by an absolute constant factor, it suffices in the worst case to perform $O(\sqrt{km})$ Newton steps with $O(k^3n^3 + k^2n^2m)$ operations per step.

Key words. interior point methods, polynomial complexity, convex programming, structural optimization

AMS subject classifications. 90C31, 90C50, 73K40

1. Introduction.

Truss topology design (TTD) problems are perhaps the most challenging in the field of structured optimization [6]. They also pose a formidable task to the mathematical programming community. To illustrate this point, consider a (two-dimensional) truss with $n = 225$ nodes in a 15×15 grid, and where, potentially, each two nodes can be connected by a bar whose thickness is a design variable. The final structure is to have maximum strength (stiffness) and to withstand two different cases of external loading conditions. This problem has $\frac{1}{2}n(n-1) = 25,200$ design variables plus $2n \times k = 900$ auxiliary displacement variables. The problem can be converted (as shown in this paper) to a convex programming problem, but one with a huge number of variables and constraints.

The aim of this paper is to develop a new algorithmic approach to the TTD problem based on polynomial time interior point methods, and to demonstrate the ability of these methods to solve TTD problems of the aforementioned dimensions. The TTD problem discussed here is an extension to the multiload case of the model discussed in the papers by Ben-Tal and Bendsøe [2] and Ben-Tal, Cocvara, and Zowe [3]. We use the same notations and assumptions here. Informally the problem is as follows. Our goal is to design a *truss*, i.e., a construction comprised of *bars* linked with each other at *nodes* from a given finite subset of the two-dimensional plane (*ground structure*) or three-dimensional space (*spatial structure*). The construction can be subject to a given number of *loadings*, i.e., collections of external forces acting at nodes. A loading causes deformation of the truss, and the latter, as a result, capacitates certain energy (*compliance*); this energy can be thought of as a measure of stiffness of the truss (the less the compliance is, the larger the stiffness is). In what follows, we measure the quality of a truss by its largest compliance over the given set of nonsimultaneous loadings (*scenarios*) and the problem is to minimize this worst-case compliance.

* Received by the editors July 15, 1992; accepted for publication (in revised form) June 28, 1993. The research for this paper was supported by The German Israel Foundation for Scientific Research and Development (G.I.F.), and The Fund for The Promotion of Research at the Technion.

† Faculty of Industrial Engineering and Management, Technion Israel Institute of Technology, Haifa 32000, Israel (ierlb01@technion.technion.ac.il).

Formally the problem is as follows. Consider a truss with m potential bars, n nodes, and k loading cases. We have the following data:

vectors, $f_j \in \mathbb{R}^n$ $(j = 1, \ldots, k)$, the external nodal force vectors (loadings);

matrices, $A_i \in \mathbb{R}^{n \times n}$ $(i = 1, \ldots, m)$, the unit bar-stiffness matrices;

vectors, $L, U \in \mathbb{R}^m$, the lower and upper bounds on bar volumes;

scalar, V, the total bar volume.

Variables of the problem are

$x_j \in \mathbb{R}^n$ $(j = 1, \ldots, k)$, vector of nodal displacements of the jth loading case (analysis variables);

$t \in \mathbb{R}^m$, vector of bar volumes (design variables).

The problem of optimal TTD is

$$\text{minimize } \psi\left(t, \{x_j\}_{j=1}^k\right) = \max_{1 \le j \le k} f_j^T x_j \tag{1.1}$$

by choice of $t \in \mathbb{R}^m, x_j \in \mathbb{R}^n$, $j = 1, \ldots, k$, *subject to the constraints*

$$A(t)x_j = f_j, \ j = 1, \ldots, k\ , \tag{1.2}$$

$$\sum_{i=1}^m t_i = V, \tag{1.3}$$

$$L \le t \le U, \tag{1.4}$$

where

$$A(t) = \sum_{i=1}^m t_i A_i\ .$$

Note that the objective (1.1) is the worst (over the k scenarios) compliance.

Now the origin of matrices A_i is the following. When formulating the problem, every node is associated with a linear space of *virtual displacements*; this is the whole plane (for the planar trusses) or space (for the spatial ones), if the node is not restricted in its movements, and the tangent space to the manifold of the allowed locations of the node for the nodes that are partially or completely fixed (in the case of a completely fixed node the space of virtual displacements is $\{0\}$). Let β_i be the cosine vector of the direction of the bar i (with respect to certain fixed orientation of the bars); then the m-dimensional vector δ of *bar elongations* is related to the load according to the equation (Young's law)

$$P_\nu(f)_\nu = P_\nu \left\{ \sum_{i \in B_{\text{in}}(\nu)} E \frac{t_i}{\ell_i^2} \beta_i \delta_i - \sum_{i \in B_{\text{out}}(\nu)} E \frac{t_i}{\ell_i^2} \beta_i \delta_i \right\}, \tag{1.5}$$

where P_ν is the orthoprojector onto the space of virtual displacements of the node ν, $(f)_\nu$ is the external force applied at the node, $B_{\text{in}}(\nu)$, $B_{\text{out}}(\nu)$ denote the sets of bars that start and end, respectively, at the node ν, t_i is the volume of ith bar, ℓ_i is the length of the bar, and E is the Young modulus.

If ith bar starts at the node $\nu'(i)$ and ends at the node $\nu(i)$, then one clearly has

$$\delta_i = \beta_i^T \left(\xi_{\nu(i)} - \xi_{\nu'(i)}\right),$$

where ξ_ν is the component of the displacement vector x associated with νth node. We conclude that

(1.6)

$$P_\nu(f)_\nu = P_\nu \left\{ \sum_{i\in B_{\text{in}}(\nu)} E\,\frac{t_i}{\ell_i^2}\,\beta_i\beta_i^T\left(\xi_\nu - \xi_{\nu'(i)}\right) - \sum_{i\in B_{\text{out}}(\nu)} E\,\frac{t_i}{\ell_i^2}\,\beta_i\beta_i^T\left(\xi_\nu - \xi_{\nu'(i)} - \xi_\nu\right) \right\}$$

$$= P_\nu \left\{ - \sum_{i\in B(\nu)} E\,\frac{t_i}{\ell_i^2}\,\beta_i\beta_i^T\left(\xi_\nu - \xi_{\nu'(i)}\right) \right\},$$

where $B(\nu) = B_{\text{in}}(\nu) \cup B_{\text{out}}(\nu)$.

Now, let b_i be an n-dimensional vector defined as follows. The coordinates of all our n-dimensional vectors $(x, f, \ldots)$ are in natural correspondence with the nodes of the truss and each node ν is associated with d_ν coordinates, d_ν being the number of the degrees of freedom of the node. The vector b_i has all zero coordinates, excluding those associated with the nodes $\nu(i)$ and $\nu'(i)$; the $\nu(i)$-component of b_i forms the vector $\ell_i^{-1}\sqrt{E}P_{\nu(i)}\{\beta_i\}$, while the $\nu'(i)$-component of b_i is $-\ell_i^{-1}\sqrt{E}P_{\nu'(i)}\{\beta_i\}$.

It follows from (1.6) that the equilibium equation for truss is

$$f = A(t)x,$$

where, as it was claimed,

$$A(t) = \sum_{i=1}^m t_i A_i, \qquad A_i = b_i b_i^T. \tag{1.7}$$

In our theoretical considerations, we are not always using all properties of the matrices A_i that follow from (1.7). What is crucial and henceforth assumed throughout the paper is the following consequence of (1.7).

A.1. All A_i are symmetric positive semidefinite $n \times n$ matrices. In some parts of the text, this will be the only assumption concerning the nature of the matrices A_i. Sometimes (mainly to simplify matters) we shall also use the following fact.

A.1$_1$. All A_i are dyadic: $A_i = b_i b_i^T$. This assumption is referred to as the *dyadic case.*

Besides A.1, throughout the paper we use the following nondegeneracy assumptions (that clearly are not restrictive for the truss problem).

A.2. The matrices A_i have no common nontrivial null directions, i.e., $A(t)$ is positive definite for $t > 0$.

A.3. The linear constraints $L \le t \le U$, $\sum_{i=1}^m t_i = V$ are admissible in the sense that

$$0 \le L < U; \qquad \sum_{i=1}^m L_i < V < \sum_{i=1}^m U_i,$$

and all loadings f_j are nonzero vectors.

By allowing the bar volumes to reduce to zero, we allow removal of bars from the ground structure, thus designing the *truss topology* (connectivity pattern). Only the locations of the nodes are predetermined. However, by choosing a dense grid for this ground structure, and thus a large number of nodes, we also capture in our formulation

of the TTD problem the *geometry design* (nodal locations) aspect of the problem. If there are n nodes, and each one of them is connected to all other nodes, then the number of potential bars is $m = \frac{1}{2}n(n-1)$. Thus the TTD problem falls into the realm of *large-scale* nonlinear optimization problems, and it is here where polynomial-time algorithms become crucially important. It turns out that the TTD problem can be reformulated as a *convex* programming problem; although highly nonlinear, this problem possesses nice analytical structure and, therefore, can be solved by interior point polynomial time methods. This possibility was already studied in Ben-Tal and Nemirovskii [5], where a *path-following* method is employed. In this paper we present a *potential reduction* polynomial primal-dual algorithm for the TTD that has the same worst-case theoretical efficiency as the path-following method from the latter paper, but that demonstrates a much better practical behaviour.

The linear programming (LP) prototype of the method used in this paper was developed by Todd and Ye [11] and Ye [12]; a nonlinear extension of the method was given in Nesterov and Nemirovskii [8]–[10]. To apply this method to the TTD problem, we first appropriately reformulate the problem; this reformulation (which is of interest in its own right) forms the subject of §1. Section 2 contains further transformation of the problem required by the potential reduction machinery. The method itself is represented in §3, and §4 contains some numerical results.

2. "Bar volumes-bar forces" reformulation of the TTD. To apply a potential reduction interior point method to the TTD problem, we first embed the problem into the so-called "conic" form

$$(P) \qquad \text{minimize } c^T y \quad \text{s.t.} \quad y \in \mathcal{K} \cap \mathcal{L},$$

where $\mathcal{K}$ is a closed convex cone in certain $\mathcal{E} = \mathbb{R}^N$ and $\mathcal{L}$ is an affine subspace in $\mathcal{E}$. Note that (P) is similar to the standard LP problem; the only difference is that in a general conic problem, $\mathcal{K}$ should not be a polyhedral cone. In the case of TTD this cone turns out to be a direct product of the nonnegative orthant and a number of standard second order cones (see below), so that the resulting (P) is an "almost" quadratically constrained quadratic programming problem.

We start with the following observation. Let $Q_0, \ldots, Q_p$ be positive semidefinite symmetric $r \times r$ matrices with positive definite sum; these matrices define the matrix-valued function

$$Q(w) = Q_0 + \sum_{i=1}^{p} w_i Q_i$$

on $\mathbb{R}^p$. Also let $g(w)$ be an affine mapping from $\mathbb{R}^p$ into $\mathbb{R}^r$. The functions $g(w)$, $Q(w)$ define the quadratic form of $x \in \mathbb{R}^r$ (this form depends on w as on a parameter)

$$S_w(x) = 2g^T(w)x - x^T Q(w)x \ .$$

This quadratic form is concave for $w \geq 0$ and strictly concave for $w > 0$ (recall that Q_i are positive semidefinite and their sum is positive definite). It follows that the function

$$F(w) = \sup\{S_w(x) | x \in \mathbb{R}^r\} : \ \mathbb{R}^p_+ \to \mathbb{R} \cup \{+\infty\}$$

(which is clearly convex and lower semicontinuous) is finite at least on the interior $\mathbb{R}^p_{++}$ of the nonnegative orthant, where $F(w)$ is simply the quadratic-fractional function $g^T(w)Q^{-1}(w)g(w)$. Let us derive certain special representation of the epigraph

$$\mathcal{G} = \{(w,\tau) \in \mathbb{R}^p \times \mathbb{R} \,|\, w \geq 0, \quad \tau \geq F(w)\}$$

of this function. To this end, let us find $\ell_i \times r$ matrices B_i in such a way that

$$Q_i = B_i^T B_i, \qquad i = 0, \ldots, p,$$

and let q denote the vector variable with the components $q_0, \ldots, q_p$ of the dimensions $\ell_0, \ldots, \ell_p$. Our goal is to prove the following lemma.

LEMMA 1. *A pair (w,t) belongs to $\mathcal{G}$ if and only if there exists a vector q of the above structure which satisfies the linear (vector) equation*

$$\sum_{i=0}^{p} B_i^T q_i = g(w) \tag{2.1}$$

and the inequality

$$\|q_0\|_2^2 + \sum_{i=1}^{p} \|q_i\|_2^2 / w_i \leq \tau. \tag{2.2}$$

(From now on the value of a fraction $a/0$ is 0 if $a = 0$ and $+\infty$ if $a > 0$; the case of $a < 0$ will not occur.) □

Proof. The only if part. Let $(w,\tau) \in \mathcal{G}$, so that $w \geq 0$ and the concave quadratic form $S_w(\cdot)$ is bounded above. Then this form attains its maximum at certain $x \in \mathbb{R}^r$, and this x is a solution to the system

$$Q(w)x = g(w)\ ; \tag{2.3}$$

note that

$$F(w) = g^T(w)x\ . \tag{2.4}$$

Relation (2.3) means that

$$B_0^T(B_0 x) + \sum_{i=1}^{p} B_i^T(w_i B_i x) = g(w)\ , \tag{2.5}$$

so that the vectors

$$q_0 = B_0 x;\ q_i = w_i B_i x, \qquad i = 1, \ldots, p$$

determine a vector $q = (q_0, \ldots, q_p)$ satisfying (2.1). Multiplying (2.5) by x^T and taking into account (2.4), we find that

$$F(w) = \|q_0\|_2^2 + \sum_{i=1}^{p} \|q_i\|_2^2 / w_i\ ,$$

and (2.2) follows.

The if part. Let q, τ and $w \geq 0$ satisfy (2.1), (2.2). We prove that $F(w)$ is well defined and is $\leq \tau$. Let I be the set of all $i \in \{1, 2, \ldots, p\}$ such that $w_i > 0$. It is possible that $w = 0$, i.e., I is empty; then (2.2), according to our rule for assigning values to fractions $a/0$, means that $q_i = 0$, $i = 1, \ldots, p$, whence, in view of (2.1), $B_0^T q_0 = g(0)$. Let $\bar{q}_0$ be the normal (i.e., of the minimal $\|\cdot\|_2$-norm) solution to the latter system; then $\bar{q}_0 = B_0 x$ for some x and $\|\bar{q}_0\|_2^2 \leq \|q_0\|_2^2 \leq \tau$. In other words, we have

$$Q(0)x = Q_0 x = B_0^T B_0 x = B_0^T \bar{q}_0 = g(0)$$

and

$$g^T(0)x = \left(B_0^T \bar{q}_0\right)^T x = (\bar{q}_0)^T \bar{q}_0 \leq \tau \ ,$$

so that $F(0) \leq \tau$, as claimed.

Now, let $w \neq 0$, so that I is not empty. Just as above, we have $q_i = 0$ for $i \notin E$, $i \geq 1$, so that

$$B_0^T q_0 + \sum_{i \in I} B_i^T q_i = g(w) \ .$$

It follows that the problem

$$\text{(2.6)} \qquad \text{minimize } \|r_0\|_2^2 + \sum_{i \in I} \|r_i\|_2^2 / w_i \text{ s.t. } B_0^T r_0 + \sum_{i \in I} B_i^T r_i = g(w)$$

with the control variables r_0, $\{r_i,\ i \in I\}$ is feasible and therefore, of course, solvable; let $\bar{q}_0$, $\{\bar{q}_i,\ i \in I\}$ be the optimal solution to the problem. Note that

$$\text{(2.7)} \qquad \|\bar{q}_0\|_2^2 + \sum_{i \in I} \|\bar{q}_i\|_2^2 / w_i \leq \|q_0\|_2^2 + \sum_{i \in I} \|q_i\|_2^2 / w_i \leq \tau \ .$$

From optimality condition for (2.6), it follows that there exists $x \in \mathbb{R}^r$ (a vector of Lagrange multipliers for the linear constraints in (2.6)) such that

$$\bar{q}_0 = B_0 x, \quad \bar{q}_i = w_i B_i x, \quad i \in I \ ;$$

in other words,

$$Q(w)x = B_0^T \bar{q}_0 + \sum_{i \in I} B_i^T \bar{q}_i = g(w)$$

and

$$g^T(w)x = \left(B_0^T \bar{q}_0\right)^T x + \sum_{i \in I} \left(B_i^T \bar{q}_i\right)^T x = \|\bar{q}_0\|_2^2 + \sum_{i \in I} \|\bar{q}_i\|_2^2 / w_i \leq \tau$$

(the latter inequality is (2.7)). Whence, by the comments at the beginning of the proof, $F(w) \leq \tau$. □

Now note that in view of A.1 and the comments from the beginning of the proof of the lemma, the TTD problem (1.1)–(1.4) can be rewritten in the following equivalent form:

$$\text{minimize } \max_{j \leq k} \tau_j \text{ s.t. } \tau_j \geq \sup_x \{2 f_j^T x - x^T A(t) x\}, \ \sum_{i=1}^m t_i = V, \ L \leq t \leq U.$$

By the lemma, the latter problem, in turn, is equivalent to

$$(\mathcal{B}) \text{ minimize } \psi(\tau, t, q) \equiv \max_{j \le k} \tau_j \tag{2.8}$$

subject to

$$\tau_j \ge \sum_{i=1}^{m} \|q_{ij}\|_2^2 / t_i, \qquad j = 1, \ldots, k, \tag{2.9}$$

$$\sum_{i=1}^{m} B_i^T q_{ij} = f_j, \qquad j = 1, \ldots, k, \tag{2.10}$$

$$\sum_{i=1}^{m} t_i = V, \qquad U \ge t \ge L. \tag{2.11}$$

Note that in the dyadic case $A_i = b_i b_i^T$, one can take $B_i = b_i^T$ so that $\ell_i = 1$, $i = 1, \ldots, m$, and q_{ij} turn out to be scalars. The quantity q_{ij} is nothing but the reaction force caused by elongation of the ith bar under the jth loading, so that (2.8)–(2.11) may be called "bar volumes-bar forces" reformulation of the TTD problem. Furthermore, in the dyadic case with *simple bounds on bar volumes*, i.e., when $L_i \equiv 0$, $U_i \equiv V$, problem (2.8)–(2.11) can be rewritten as

$$\underset{q,t}{\text{minimize}} \quad \max_{\lambda \in \Lambda} \left\{ \sum_{j=1}^{k} \lambda_j \sum_{i=1}^{m} q_{ij}^2 / t_i \right\} \tag{2.12}$$

subject to

$$\sum_{i=1}^{m} q_{ij} b_i = f_j, \qquad j = 1, \ldots, k, \tag{2.13}$$

$$t \ge 0, \qquad \sum_{i=1}^{m} t_i = V, \tag{2.14}$$

where $\Lambda = \{\lambda \in \mathbb{R}^k | \lambda \ge 0, \sum_{j=1}^{k} \lambda_j = 1\}$. One can immediately eliminate t from the latter problem, which results in the following problem in variables q:

$$\text{minimize} \quad \max_{\lambda \in \Lambda} V^{-1} \left\{ \sum_{i=1}^{m} \sqrt{\sum_{j=1}^{k} \lambda_j q_{ij}^2} \right\}^2 \tag{2.15}$$

subject to

$$\sum_{i=1}^{m} q_{ij} b_i = f_j, \qquad j = 1 \ldots, k. \tag{2.16}$$

In the single-load case $k = 1$ problem (2.15)–(2.16) is nothing but an LP program. This LP reformulation of a single-load TTD problem with simple bounds on bar volumes was found in Achtziger et al. [1].

3. Conic reformulation of the TTD problem.

3.1. Our next step is to obtain the conic reformulation of the TTD problem. To this end, we introduce additional scalar variables r_{ij}, p_{ij} subject to constraints

$$r_{ij} - p_{ij} = t_i,\ r_{ij} \geq \left\{\|q_{ij}\|_2^2 + p_{ij}^2\right\}^{1/2}. \tag{3.1}$$

Note that from (3.1) it follows that

$$r_{ij} + p_{ij} \geq \|q_{ij}\|_2^2/(r_{ij} - p_{ij}) = \|q_{ij}\|_2^2/t_i. \tag{3.2}$$

On the other hand, if (τ, t, q) is a feasible solution to (2.8)–(2.11) with finite value of the objective, we always can choose r_{ij}, p_{ij} satisfying (3.1) in such a way that the inequality in (3.2) is an equality for all i, j. We immediately conclude that (2.8)–(2.11) is equivalent to the problem

$$\text{minimize} \quad \max_{j \leq k} \tau_j \tag{3.3}$$

subject to linear constraints

$$\tau_j \geq \sum_{i=1}^{m} (r_{ij} + p_{ij}), \qquad j = 1, \ldots, k; \tag{3.4}$$

$$r_{ij} - p_{ij} = t_i, \quad i = 1, \ldots, m, \quad j = 1, \ldots, k, \tag{3.5}$$

$$\sum_{i=1}^{m} t_i = V, \qquad U \geq t \geq L, \tag{3.6}$$

$$\sum_{i=1}^{m} B_i^T q_{ij} = f_j, \qquad j = 1, \ldots, m, \tag{3.7}$$

and nonlinear constraints

$$r_{ij} \geq \left\{\|q_{ij}\|_2^2 + p_{ij}^2\right\}^{1/2}, \quad i = 1, \ldots, m, \quad j = 1, \ldots, k. \tag{3.8}$$

To bring the problem to the conic formulation, we need a linear objective function, linear equality constraints, and a cone constraint. To this end we represent t_i as convex combinations $U_i u_i + L_i w_i$ of L_i and U_i, the coefficients of the combinations being our new variables, and add slack variables v_j to convert (3.3)–(3.8) to the problem

$$(TD^p): \quad \text{minimize} \quad \phi(y) \equiv c^T y = k^{-1} \left\{ \sum_{j=1}^{k} \left[\sum_{i=1}^{m} (r_{ij} + p_{ij}) + v_j \right] \right\} \tag{3.9}$$

by choice of $y \equiv (v_1, \ldots, v_k \in \mathbb{R};\ u_1, \ldots, u_m,\ w_1, \ldots, w_m \in \mathbb{R};\ p_{ij}, r_{ij} \in \mathbb{R},\ q_{ij} \in \mathbb{R}^{\ell_i},\ i = 1, \ldots, m,\ j = 1, \ldots, k)$ subject to the constraints

$$\sum_{i=1}^{m} (r_{ij} + p_{ij}) + v_j = \sum_{i=1}^{m} (r_{i1} + p_{i1}) + v_1, \qquad j = 2, \ldots, k, \tag{3.10}$$

$$r_{ij} - p_{ij} = U_i u_i + L_i w_i, \quad i = 1, \ldots, m, \quad j = 1, \ldots, k, \tag{3.11}$$

$$u_i + w_i = 1, \qquad i = 1, \ldots, m, \tag{3.12}$$

$$\sum_{i=1}^{m} B_i^T q_{ij} = f_j\, j, \qquad j = 1, \ldots, m, \tag{3.13}$$

$$\sum_{i=1}^{m} \{U_i u_i + L_i w_i\} = V, \tag{3.14}$$

$$u_i \geq 0,\ w_i \geq 0, \quad i = 1, \ldots, m; \quad v_j \geq 0, \quad j = 1, \ldots, k;$$
$$r_{ij} \geq \left\{ \|q_{ij}\|_2^2 + p_{ij}^2 \right\}^{1/2}, \quad i = 1, \ldots, m, \quad j = 1, \ldots, k. \tag{3.15}$$

Now, let $\mathcal{E}$ be the space of vectors

$$y = (u, w \in \mathbb{R}^m;\ v \in \mathbb{R}^k;\ (p_{ij}, q_{ij}, r_{ij}) \in \mathbb{R}^{\ell_i + 2},\ i = 1, \ldots, m,\ j = 1, \ldots, k), \tag{3.16}$$

and let $\mathcal{K}$ be the following cone in $\mathcal{E}$:

$$\begin{aligned} \mathcal{K} = \{ & y = (u, w; v; p., q., r.) \in \mathcal{E} \,|\, u \geq 0,\ w \geq 0,\ v \geq 0, \\ & (p_{ij}, q_{ij}, r_{ij}) \in K_{ij},\ \ i = 1, \ldots, m,\ j = 1, \ldots, k\}, \end{aligned} \tag{3.17}$$

where K_{ij} is the standard second-order cone in $\mathbb{R}^{\ell_i + 2}$:

$$K_{ij} = \left\{ (p_{ij}, q_{ij}, r_{ij}) \,|\, r_{ij} \geq \|(p_{ij}, q_{ij})\|_2 \right\}. \tag{3.18}$$

Also let $\mathcal{L}$ be the affine plane in $\mathcal{E}$ defined as the set of all $y = (u, w; v; p., q., r.) \in \mathcal{E}$ satisfying the linear system (3.10)–(3.14).

It is clear that $\mathcal{K}$ is a closed convex pointed ($\mathcal{K} \cap \{-\mathcal{K}\} = \{0\}$) cone with a nonempty interior in $\mathcal{E}$, and that the constraints (3.10)–(3.15) mean precisely that $y \in \mathcal{L} \cap \mathcal{K}$. Thus, problem (TD^p) is exactly the desired conic form $(\mathcal{P})$.

It is worth noting that $\mathcal{K}$ is a direct product of a nonnegative orthant $\mathbb{R}_+^{2m+k}$ and mk standard second-order cones K_{ij}, so that the resulting problem $(\mathcal{P})$ is basically a quadratically constrained convex quadratic problem (QCQP). More precisely, the feasible set $\mathcal{L} \cap \mathcal{K}$ of $(\mathcal{P})$ in our case is the intersection of a number of sets, each of the sets being defined by a linear or a quadratic constraint. Although these quadratic constraints may be nonconvex (this is the reason we use the term "basically" in the previous sentence), the sets defined by the constraints are convex. As far as potential reduction interior point machinery is concerned, the problems of this type are no worse than those involving linear and convex quadratic constraints.

Remark. Following the same line of argument as above, one can also obtain QCQP reformulation for the TTD problem with *self-weight loads.* In the latter problem, the total nodal force associated with the jth scenario is $f_j(t) = f_j + \sum_{i=1}^m t_i g_i$, g_i being the specific nodal gravitational force vector due to the self-weight of the ith bar. Thus, the approach developed in this paper can be applied to the TTD problem with self-weight loads as well.

3.2. Our final step is to form the *Fenchel dual* to (TD^p). It is easily seen that Fenchel duality, as applied to a conic problem

$$(\mathcal{P}) \qquad \text{minimize } c^T y \quad \text{s.t. } y \in \mathcal{K} \cap \mathcal{L},$$

($\mathcal{K}$ is a closed convex cone in $\mathcal{E} = \mathbb{R}^N$; $\mathcal{L}$ is an affine subspace of $\mathcal{E}$), results in the following dual problem:

$$(\mathcal{D}) \qquad \text{minimize } b^T\eta \quad \text{s.t. } \eta \in \mathcal{K}^* \cap \mathcal{L}^*.$$

Herein the *translation vector* b is an arbitrary element of $\mathcal{L}$ (various choices of b result in equivalent objectives, i.e., those differing at the feasible set of $(\mathcal{D})$ by an additive constant), $\mathcal{K}^*$ is the dual to $\mathcal{K}$ cone, i.e.,

$$\mathcal{K}^* \equiv \{s \in \mathcal{E} \mid s^T y \geq 0,\ y \in \mathcal{K}\},$$

and

$$\mathcal{L}^* = \mathcal{M}^\perp + c,$$

$\mathcal{M}^\perp$ being the annulator of the linear space $\mathcal{M} \equiv \mathcal{L} - b$, i.e.,

$$\mathcal{M}^\perp = \{s \mid s^T(y - b) = 0,\ y \in \mathcal{L}\}.$$

Note that $\mathcal{L} = \mathcal{M} + b$, so that the duality is completely symmetric. The essence of this duality is given by the following statement.

PROPOSITION 1. *Let the cone $\mathcal{K}$ be closed, convex, pointed and with a nonempty interior, and let both of the problems $(\mathcal{P})$, $(\mathcal{D})$ be* strictly feasible *(i.e., let them possess feasible solutions belonging to the interiors of the corresponding cones). Then, both problems are solvable, and for their optimal values one has*

$$\mathcal{P}^* + \mathcal{D}^* = c^T b, \tag{3.19}$$

while for any pair (y, η) of feasible solutions to the problems one has

$$c^T y + b^T \eta = c^T b + \eta^T y \geq c^T b. \tag{3.20}$$

In particular, a pair (y, η) comprised of feasible solutions to the problems is comprised of optimal solutions if and only if

$$c^T y + b^T \eta - c^T b = 0 \tag{3.21}$$

(zero duality gap) or, which is the same, if and only if

$$\eta^T y = 0 \tag{3.22}$$

(complementary slackness).

The proof of this statement follows as a particular case of general results on Fenchel duality. It can be also found in [10].

Let us apply the above scheme to the conic reformulation (TD^p) of the TTD problem. To simplify notation, from now on we restrict ourselves to the dyadic case (which is the relevant case for the TTD), so that in (TD^p) $\ell_i = 1$, $B_i = b_i^T$, and q_{ij} are scalars. A straightforward application of the above duality scheme to (TD^p) results in the following.

Let $\mathcal{F}$ be the space of vectors

$$\omega = (\lambda_j \in \mathbb{R},\ j = 1, \ldots, k;\ v_i \in \mathbb{R},\ i = 1, \ldots, m$$
$$\gamma_{ij} \in \mathbb{R},\ i = 1, \ldots, m,\ j = 1, \ldots, k;\ \xi_j \in \mathbb{R}^n,\ j = 1, \ldots, k;\ \sigma \in \mathbb{R}),$$

and let $\mathcal{A} : \mathcal{F} \to \mathcal{E}$ be the linear mapping defined as follows:

(3.23)

$$[\mathcal{A}\omega]_{v_j} = \lambda_j;\ [\mathcal{A}\omega]_{u_i} = v_i + U_i\left(\sigma - \sum_{j=1}^{k}\gamma_{ij}\right);\ [\mathcal{A}\omega]_{w_i} = v_i + L_i\left(\sigma - \sum_{j=1}^{k}\gamma_{ij}\right),$$
$$[\mathcal{A}\omega]_{p_{ij}} = \lambda_j - \gamma_{ij};\ [\mathcal{A}\omega]_{r_{ij}} = \lambda_j + \gamma_{ij};$$
$$[\mathcal{A}\omega]_{q_{ij}} = b_i^T\xi_j.$$

Then the affine space $\mathcal{L}^*$ in the Fenchel dual to (TD^p) is

$$\mathcal{L}^* = \left\{\eta = \mathcal{A}\omega \,|\, \omega = (\lambda_\cdot, v_\cdot, \gamma_{\cdot,\cdot}, \xi_\cdot, \sigma) \in \mathcal{F} : \sum_{j=1}^{k}\lambda_j = 1\right\}, \tag{3.24}$$

the cone $\mathcal{K}^*$ dual to $\mathcal{K}$ is $\mathcal{K}$ itself, so that the problem (TD^d) that is Fenchel dual to (TD^p) is

$$(TD^d): \quad \text{minimize } \psi(\eta) \equiv \beta^T\eta \tag{3.25}$$

by choice of $\eta \in \mathcal{E}$ to the constraints

$$\eta = \mathcal{A}\omega \text{ for some } \omega \in \mathcal{F}_0 \equiv \left\{\omega = (\lambda_\cdot, v_\cdot, \gamma_{\cdot,\cdot}, \xi_\cdot, \sigma) \in \mathcal{F} : \sum_{j=1}^{k}\lambda_j = 1\right\}, \tag{3.26}$$

$$\eta \in \mathcal{K}\,. \tag{3.27}$$

Herein $\beta \in \mathcal{E}$ is a particular solution to the linear system (3.11)–(3.14).

4. The primal-dual potential reduction method for TTD. In this section, we describe a potential reduction polynomial time method for TTD. The LP prototype of the method was developed in [11] and [12] and nonlinear extensions were given in [7]–[10]. The method deals with the primal-dual pair of conic problems (TD^p), (TD^d). In the description below, one of the problems is called primal (P) and the other is called dual (D).

From now on we denote the feasible plane, the objective and the translation vector involved in (P) by $\mathcal{L}, c$, and b, respectively, while the feasible plane for (D) is denoted $\mathcal{L}^*$. Recall that above $\mathcal{L}, b$, and c were the data associated with TD^p) so that our previous notation coincides with the new one only in the case when (P) is (TD^p), not (TD^d).

4.1. A barrier for $\mathcal{K}$. The method is associated with the logarithmic barrier (interior penalty function) for the cone $\mathcal{K}$ given by (3.17):

$$\begin{aligned}\mathrm{B}(y) = &-\sum_{j=1}^{k}\ell n\, v_i - \sum_{i=1}^{m}\ell n\, u_i - \sum_{i=1}^{m}\ell n\, w_i \\ &-\sum_{i=1}^{m}\sum_{j=1}^{k}\ell n\,\left(r_{ij}^2 - p_{ij}^2 - q_{ij}^2\right) : \text{ int } \mathcal{K} \to \mathbb{R}.\end{aligned} \tag{4.1}$$

The properties of the barrier that are crucial for what follows are:

self-concordance,

$$|D^3\mathrm{B}(y)[h,h,h]| \le 2\{h^T\mathrm{B}''(y)h\}^{3/2}, \quad y \in \text{ int } \mathcal{K}, \quad h \in \mathcal{E};$$

barrier property,

$\mathrm{B}(y_i) \to \infty$ whenever $y_i \in \mathcal{K}' = \text{int}\,\mathcal{K}$ converge to a boundary point of $\mathcal{K}$;

logarithmic homogeneity,

$$\mathrm{B}(sy) = \mathrm{B}(y) - \theta\,\ell n\, s, \quad y \in \mathcal{K}', \quad s > 0,$$

where

$$\theta = 2m + k + 2mk.$$

The next (and last) property is not so crucial, although it is responsible for the particular form of the primal-dual method presented below.

self-duality, $\mathrm{B}(-\eta)$ is, up to an additive constant, the Legendre transformation of B,

$$\mathrm{B}(-\eta) = \sup\{\eta^T y - \mathrm{B}(y) | y \in \mathcal{K}'\} + \theta - 2mk\,\ell n\, 2.$$

The verification of these properties for $\mathrm{B}(y)$ as given in (4.1) can be found in Nesterov and Nemirovskii [8]–[10], where one also can find clarification of the role played by the properties in constructing polynomial time algorithms.

4.2. The primal-dual method. This method, called $PD(\mu)$ is defined in terms of the control parameter $\mu > 0$, and is associated with the *potential*

$$W(y,\eta) = \mathrm{B}(y) + \mathrm{B}(\eta) + \gamma \ell n(\eta^T y), \qquad \gamma \equiv \theta + \mu\sqrt{\theta}. \tag{4.2}$$

$PD(\mu)$ is comprised of an *initialization phase* and a *main phase*.

4.2.1. Initialization. At this phase, one finds a pair (y_0, η_0) of *strictly feasible* primal-dual solutions. In the TTD case, one can choose as a strictly feasible solution to (TD^d), say, the point $\mathcal{A}\bar{\omega}$,

$$\bar{\omega} = \{\lambda_j \equiv 1/k;\ v_i \equiv 1;\ \gamma_{ij} \equiv 1/k;\ \xi_j \equiv 0;\ \sigma = 1\}$$

(see (3.23)), while a strictly feasible solution to (TD^p) can be chosen as follows. First, we find $t_i, L_i < t_i < U_i$, in such a way that $\sum_{i=1}^m t_i = V$ (this is possible in view of A.3) and then we represent t_i as a convex combination of U_i and L_i, that results in positive u_i and w_i satisfying (3.12), (3.14). Second, we find q_{ij} satisfying (3.13), say, set

$$q_{ij} = t_i b_i^T \{t_i A_1 + \cdots + t_m A_m\}^{-1} f_j.$$

After t_i and q_{ij} are chosen, we find p_{ij} and r_{ij} satisfying the relations

$$r_{ij} - p_{ij} = t_i, \quad r_{ij} + p_{ij} > q_{ij}/t_i,$$

which ensures (3.11) and the inequalities required by (3.15)

$$r_{ij} > \{q_{ij}^2 + p_{ij}^2\}^{1/2}.$$

Now it is quite straightforward to provide (3.10) by choice of positive v_j, which completes the computation of a strictly feasible solution to (TD^p).

4.2.2. Main phase. At this phase, we form the iterates (y_p, η_p), $p = 0, 1, \ldots$ of the strictly feasible primal-dual pair found during the initialization; all of these iterates are also strictly feasible, and we ensure a reduction of the potential at each step by at least a certain positive quantity $\Omega(\mu)$ depending on μ only.

At the pth step, we first apply to the current strictly feasible primal-dual pair $(y, \eta) \equiv (y_{p-1}, \eta_{p-1})$ the *basic updating rule* $\mathcal{U}$ that transforms the pair into a new strictly feasible primal-dual pair (y^+, η^+) in such a way that

$$W(y^+, \eta^+) \leq W(y, \eta) - \Omega(\mu) \ . \tag{4.3}$$

In the basic version of the method, one sets $(y_p, \eta_p) = (y^+, \eta^+)$; in more advanced versions we may also make an attempt to improve (y^+, η^+) (i.e., to find another strictly feasible pair with better value of the potential) and take as (y_p, η_p) this improved pair.

It remains to specify the basic updating rule $\mathcal{U}$ as follows. Given a strictly feasible primal-dual pair (y, η),

1. compute the vectors

$$\xi = \mathrm{B}'(y) + \gamma(\eta^T y)^{-1}\eta \ ,$$
$$\chi = (\mathrm{B}''(y))^{-1}\xi \ ,$$

and the projection ζ, of the vector χ on the linear subspace $\mathcal{L} - b$, the projection being taken with respect to the Euclidean metric $\|h\|_{\mathrm{B}''(y)} \equiv \{h^T \mathrm{B}''(y) h\}^{1/2}$; set also

$$\xi^{\perp} = -\mathrm{B}'(y) + \mathrm{B}''(y)\zeta \ .$$

2. Set

$$\lambda = (\zeta^T \mathrm{B}''(y)\zeta)^{1/2},$$
$$y^{\#} = y - (1+\lambda)^{-1}\zeta \ ,$$
$$\eta^{\#} = \gamma^{-1}(\eta^T y)\xi^{\perp}.$$

3. The point $y^{\#}$ surely belongs to int $\mathcal{K}$. Check whether $\eta^{\#}$ also belongs to int $\mathcal{K}$; if not, set

$$(y^+, \eta^+) = (y^{\#}, \eta) \ ;$$

otherwise, choose from the two pairs $(y^{\#}, \eta), (y, \eta^{\#})$ the one with the smaller value of the potential W, and take it as (y^+, η^+).

4.2.3. Justification. The above construction is justified by the following statements [8]–[10].

PROPOSITION 2. (i) *Let (y, η) be a strictly feasible primal-dual pair. Then its iterate (y^+, η^+) is also strictly primal-dual feasible, and*

$$W(y^+, \eta^+) \leq W(y, \eta) - \Omega(\mu) \ , \tag{4.4}$$

where

$$\Omega(\mu) = \max_{\delta \in (0,1)} \min\{\mu^2/(1 + \mu\theta^{-1/2}) + \delta + \ell n(1-\delta) - \mu\delta; \ \delta - \ell n(1+\delta)\}. \tag{4.5}$$

(ii) *For an arbitrary strictly feasible primal-dual pair (y, η) one has*

$$\begin{aligned} &\{c^T y - \mathcal{P}^*\} + \{\eta^T B - \mathcal{D}^*\} = \eta^T y \\ &\leq (\eta_0^T y_0)\mathrm{R}(y_0, \eta_0)\exp\{-\mu^{-1}\theta^{-1/2}(W(y_0, \eta_0) - W(y, \eta))\}, \end{aligned} \tag{4.6}$$

where $\mathcal{P}^*$ and $\mathcal{D}^*$ are the optimal values of $(\mathcal{P})$, $(\mathcal{D})$, respectively,

$$\theta = 2m + k + 2mk$$

is the homogeneity parameter of the barrier B *and*

$$\text{(4.7)}\quad \mathrm{R}(y_0, \eta_0) = \exp\left\{\mu^{-1}\theta^{-1/2}(\mathrm{B}(y_0) + \mathrm{B}(\eta_0) + \theta\ell n(\eta_0^T y_0) - \theta\,\ell n\,\theta + 2mk\,\ell n 2)\right\}.$$

COROLLARY 1. *At the pth iteration of the method $PD(\mu)$, the following efficiency estimate holds for the strictly feasible primal-dual sequence $\{(y_\ell, \eta_\ell)\}_{\ell=0,1,\ldots}$*

$$\text{(4.8)}\quad \left\{c^T y_p - \mathcal{P}^*\right\} + \left\{b^T \eta_p - \mathcal{D}^*\right\} \le (\eta_0^T y_0)\mathrm{R}(y_0, \eta_0)\exp\left\{-\mu^{-1}\theta^{-1/2}\Omega(\mu)p\right\}.$$

Note that the efficiency estimate (4.6) is expressed in terms of the quantity by which the potential is reduced up to the current moment. Therefore, we aim at decreasing the potential as much as possible. The basic updating rule ensures at least $\Omega(\mu)$-reduction at each step. Any further reduction of the potential (say, by performing a line search in the directions $y^\# - y$ and $s^\# - s$) can only improve the accuracy bound. This possibility to perform *large steps* is the main practical advantage of a potential reduction method as compared to other polynomial time interior point procedures.

4.2.4. Implementation. Implementation of the primal-dual method depends basically on two issues: What are the tactics for large steps and how are the linear algebra routines implemented?

Large steps. The simplest way to perform large steps is as follows. When applying the basic updating rule to the current strictly feasible primal-dual pair (y, η), we find two directions in the space $\mathcal{E} \times \mathcal{E}$ of primal-dual variables, namely, the primal direction $d = (y^\# - y, 0)$ and the dual direction $d' = (0, \eta^\# - \eta)$. Both of these directions are primal-dual feasible, i.e., they belong to $(\mathcal{L} - b) \times (\mathcal{L}^* - c)$. Now, let L be the two-dimensional affine subspace $(y, \eta) + \mathbb{R}d + \mathbb{R}d'$ in the primal-dual feasible plane $\mathcal{L} \times \mathcal{L}^*$, and let

$$S = L \cap (\mathcal{K} \times \mathcal{K})'$$

be the intersection of the two-dimensional plane with the set of strictly feasible primal-dual solutions. To simplify the explanation, we extend the potential onto $L \setminus S$ by the value $+\infty$, so that the potential now is defined on the whole L. We know that at one of the points $(y, \eta) + d$, $(y, \eta) + d'$ the value of W is less than the one at (y, η) by at least $\Omega(\mu)$. Now we apply to the reduction of W onto L a relaxation minimization method starting with the best (i.e., with smaller value of W) of the above two points, and taking the resulting pair (which surely is at least as good as the one given by the basic updating rule) as our new strictly feasible primal-dual pair. This is a very natural scheme and the only problem is which minimization method to choose. We must take into account that W is not well defined everywhere on L and it is not clear whether the function is convex. The simplest way to resolve these difficulties is as follows. At a point $(r, s) \in S$, the potential is nothing more than $\mathrm{B}(s) + \mathrm{B}(r) + \gamma\,\ell n(\eta^T y + (s - \eta)^T b + c^T(r - y))$ (see (3.20)). Thus, at a point $(p, q) \in Q \equiv \{(p, q) \in \mathbb{R}^2 | (y, \eta) + pd + qd' \in S\}$ one has

$$w(p, q) \equiv W((y, \eta) + pd + qd') = w_0(p, q) + \gamma\,\ell n(a + bp + cq),$$

where $w_0 = \mathrm{B}(y+p(y^{\#}-y))+\mathrm{B}(\eta+q(\eta^{\#}-\eta))$. Due to its origin, w_0 is *self-concordant* [7], [9], [10], and from the results of the latter papers, it follows that the recurrence

$$(p,q)^T| \to (p,q)^T - (1+\lambda)^{-1}[w_0''(p,q)]^{-1}\nabla w(p,q)\,,$$

$$\lambda \equiv \lambda(p,q) = \{(\nabla w(p,q))^T[w_0''(p,q)]^{-1}\nabla w(p,q)\}^{1/2} \tag{4.9}$$

keeps that iterate within Q and decreases w at least by $\{\lambda - \ell n(1+\lambda)\}$. We can use this recurrence as the required relaxation algorithm for w. Note that a step of this recurrence is relatively cheap (we deal with a two-dimensional problem, and it is easily seen that it takes no more than $O(mk)$ operations to update the current (p,q)). Thus, a reasonable tactic for large steps could be to perform a fixed (say, 20) steps of the recurrence (4.9), starting with the pair given by the basic updating rule.

Linear algebra. An iteration of the above method consists mainly of computing the projection (with respect to the Euclidean structure defined by $\mathrm{B}''(y)$) of a given vector onto $\mathcal{L}$; all remaining computations (including the above implementation of large steps) are quite straightforward and require $O(mk)$ arithmetic operations.

How is the above projection computed? This depends on which one of the problems—(TD^p) or (TD^q)—one considers as $(\mathcal{P})$. Let us start with the case when $(\mathcal{P})$ is (TD^d), so that

$$\mathcal{L} - b \equiv \mathcal{M}_d = \left\{ \mathcal{A}\omega \,|\, \omega = (\lambda_\cdot, v_\cdot, \gamma_{\cdot,\cdot}, \xi_\cdot, \sigma) \in \mathcal{F} : \sum_{j=1}^{k} \lambda_j = 0 \right\}.$$

The projection ζ of a given vector $\chi \in \mathcal{E}$ onto $\mathcal{L} - b$ in this case is

$$\zeta = \mathcal{A}\omega,$$

where $\omega \in \mathcal{F}$ is defined by the linear system with unknowns $\omega \in \mathcal{F}$, $r \in \mathbb{R}$:

$$\mathcal{B}\omega = \mathcal{A}^T\xi + re_k, \qquad e_k^T\omega = 0,$$

with $\mathcal{B} = \mathcal{A}^T\mathrm{B}''(y)\mathcal{A}$ and e_k being the element of $\mathcal{F}$ with λ-entries equal to one and the remaining entries equal to zero.

Solving the latter system is, basically, the same as solving a pair of (differing by the values of the right-hand side vectors) linear systems of the type

$$\mathcal{B}x = \beta. \tag{4.10}$$

Now, although $\mathcal{B}$ is of a very large size (i.e., $M \times M$, $M = \dim \mathcal{F} = (m+n+1)k+m+1$), this matrix is of extremely simple structure, i.e., its $(mk+m) \times (mk+m)$ block Q associated with the variables m_i, γ_{ij} is a block-diagonal matrix comprised of m diagonal $(k+1) \times (k+1)$ blocks Q_i, $i = 1, \ldots, m$ (Q_i is associated with the variables v_i, γ_{ij}, $j = 1, \ldots, k$). Furthermore, Q_i is a correction of the rank ≤ 4 of a diagonal matrix. Thus, we have no difficulty inverting Q and can therefore reduce solving (4.10) to assembling a "small" ($[(n+1)k+1] \times [(n+1)k+1]$) Shur complement to Q in $\mathcal{B}$ and inverting this complement. More detailed analysis demonstrates that system (4.10) can be solved at the arithmetic cost

$$c = O(m\nu^2k^2 + n^3k^3)\,, \tag{4.11}$$

ν being the maximum number of nonzero entries in the vectors b_i (recall that $A_i = b_i b_i^T$). Note that in TTD problem ν is a small integer (four for ground structures and six for spatial ones).

We conclude that in the case under consideration (i.e., when $(\mathcal{P})$ is (TD^d)), the arithmetic cost of a step in the primal-dual method is given by (4.11).

Now consider the case when $(\mathcal{P})$ is the problem (TD^p). Here our goal is to find the projection, ζ (with respect to the Euclidean structure defined by $\mathrm{B}''(y)$) of a vector $\chi \in \mathcal{E}$ on the orthogonal complement $(\mathcal{M}_d)^\perp$. To this end it suffices to compute the projection ψ of the vector $\bar{\chi} \equiv \mathrm{B}''(y)\chi$, on the space $\mathcal{M}_d$ with respect to the Euclidean structure defined by the matrix $[\mathrm{B}''(y)]^{-1}$, which can be done in the above-described manner, and then set

$$\zeta = \chi - [\mathrm{B}''(y)]^{-1}\psi$$

(Indeed, in view of the origin of ψ, we have $\psi \in \mathcal{M}_d$ and $[\mathrm{B}''(y)]^{-1}(\bar{\chi} - \psi) \in (\mathcal{M}_d)^\perp$; since $[\mathrm{B}''(y)]^{-1}(\bar{\chi} - \psi) = \chi - [\mathrm{B}''(y)]^{-1}\psi = \zeta$, we conclude that $\zeta \in (\mathcal{M}_d)^\perp$ and $[\mathrm{B}''(y)](\chi - \zeta) = \psi \in \mathcal{M}_d$, as required.)

Thus, the arithmetic cost of a step in the primal-dual method, $(\mathcal{P})$ being the problem (TD^p), also is given by (4.11).

4.2.5. Complexity. Let the control parameter μ of the method be fixed. Then the *Newton complexity* $\mathcal{N}$ of the method, i.e., the number of steps sufficient to improve the upper accuracy bound (4.8) by an absolute constant factor, is $O(1)\theta^{1/2}$, i.e.,

$$\mathcal{N} = 0(1)\sqrt{mk}.$$

Since the arithmetic cost of a step is given by (4.11), the *arithmetic cost per accuracy digit*, i.e., the number of arithmetic operations sufficient to improve upper accuracy bound (4.8) by an absolute constant factor, is

$$\mathcal{C} = 0(1)\,(m\nu^2k^2 + n^3k^3)\sqrt{mk}.$$

5. Numerical results. Our implementation of the primal-dual method for TTD follows the scheme presented in the previous section. We dealt with the case of simple bounds on bar volumes ($U_i \equiv V$, $L_i \equiv 0$); note that in this case one can eliminate variables w_i, v_i, which reduces slightly the dimensions of $\mathcal{E}$ and $\mathcal{F}$.

The numerical results are presented in Table 1. Computations were terminated when 0.5% accuracy was achieved, i.e., when the current duality gap $\eta^T y$ became less than $0.005|f|$, where f denotes the best value of the objective in $(\mathcal{P})$ found so far.

In our experiments, $(\mathcal{P})$ was chosen as the problem (TD^d), and the choice of the objective for this problem was such that the duality relation became $\mathcal{P}^* = -\mathcal{D}^*$, so that actual compliances are, up to the duality gap, negations of the values indicated in the table.

Computations were performed on a Sun 4 Sparc workstation; Fortran source codes were compiled with 03 optimization option.

The results of the tests indicated in the table are quite encouraging. In the *single-load case* they are better than those for most currently available methods (like SQP, GRG, or augmented Lagrangian methods). They are still worse by factor 2–3 than the best computational results for single-loaded TTD; see [4]. As far as the *multiloaded case* is concerned, no comparison can be made at this point as there is almost no computational experience with other methods for such large-scale problems.

TABLE 1
Primal-dual method on TTD *problems.*

Problem	N	M	K	QC	dim $\mathcal{F}$	dim $\mathcal{E}$	Best found value	Complexity	
								I	CPU time
T490–150	49	150	1	150	236	60	−44.99618	32	6″
			3	450	706	1503	−45.57234	51	1′25″
T63–1234	63	1234	1	1234	1348	4937	−10.17189	88	1′56″
			3	3702	4042	12343	−10.18890	99	11′18″
T81–2040	81	2040	1	2040	2186	8161	−10.12049	76	3′04″
			3	6120	6556	20403	−10.28114	236	46′04″
T121–4492	121	4492	1	4492	4714	17969	−29.82254	116	11′04″
			2	8984	9427	31446	−29.84763	317	1*h*17′59″
T171–8958	171	8958	1	8958	9264	35833	−12.79993	217	42′44″
			2	17916	18527	62708	−12.84929	600	5*h*40′01″
T225–15556	225	15556	1	15556	15978	62225	−29.67115	219	1*h*35′21″
			2	31112	31955	108894	−29.67956	600	13*h*50′54″

N, number of nodes.
M, number of potential bars.
K, number of loadings.
QC, number of second-order direct factors in $\mathcal{K}$ or, which is the same number of nonlinear constraints involved into the problem.
I, total number of steps.

REFERENCES

[1] W. ACHTZIGER, M. BENDSØE, A. BEN-TAL, AND J. ZOWE, *Equivalent displacement based formulations for maximum strength truss topology design*, Impact Computing Sci. Engrg., 4 (1992) pp. 315–345.

[2] A. BEN-TAL AND M. P. BENDSØE, *A new method for optimal truss topology design*, SIAM J. Optim., 3 (1993), pp. 322–358.

[3] A. BEN-TAL, A. COCVARA, AND J. ZOWE, *Two nonsmooth approaches to simultaneous geometry and topology design of trusses* in Topology Design of Structures, M.P. Bendsøe and C.A. Soares, eds., Kluwer Academic Publishers, Dordrecht, the Netherlands, 1993, pp. 31–42.

[4] A. BEN-TAL, M. ZIBULEVSKII, AND I. YUSEFOVICH, *Penalty Barrier Multipliers Methods for Minimax and Constrained Smooth Convex Programs*, Research Report 9/92, Optimization Laboratory, Faculty of Industrial Engineering and Management, Technion—Israel Institute of Technology, 1992.

[5] A. BEN-TAL AND A. NEMIROVSKII, *An interior point algorithm for truss topology design*, in Topology Design of Structures, M.P. Bendsøe and C.A. Soares, eds., Kluwer Academic Publishers, Dordrecht, the Netherlands, 1993, pp. 55–69.

[6] U. KIRSCH, *Optimal topologies of structures*, Appl. Mech. Rev., 42 (1989), pp. 223–239.

[7] YU. NESTEROV AND A. NEMIROVSKII, *Self-Concordant Functions and Polynomial Time Methods in Convex Programming*, Moscow, Central Economic and Mathematical Institute, USSR Acad. Sci., 1989.

[8] ———, *Logarithmically homogeneous barriers and potential based polynomial time methods in convex programming*, Research Report, Central Economic and Mathematical Institute, USSR Acad. Sci., Moscow, May 1990.

[9] ———, *Interior Point Polynomial Methods in Convex Programming: Theory and Applications*, Society for Industrial and Applied Mathematics, Philadelphia, 1993.

[10] ———, *"Conic" formulation of a convex programming problem and duality*, Optim. Software, 1 (1992), pp. 10–31.

[11] M. Y. TODD AND Y. YE, *A centered projective algorithm for linear programming*, Math. Oper. Res., 15 (1990), pp. 508–529.

[12] Y. YE, *An $O(n^3L)$ potential reduction algorithm for linear programming*, Math. Programming, 50 (1991), pp. 239–258.

SIAM J. OPTIMIZATION
Vol. 4, No. 3, pp. 613–625, August 1994
© 1994 Society for Industrial and Applied Mathematics
010

MONOTONICITY OF PRIMAL AND DUAL OBJECTIVE VALUES IN PRIMAL-DUAL INTERIOR-POINT ALGORITHMS*

SHINJI MIZUNO†, MICHAEL J. TODD‡, AND LEVENT TUNÇEL§

Abstract. Monotonicity of primal and dual objective values in the framework of primal-dual interior-point methods is studied. The primal-dual affine-scaling algorithm is monotone in both objectives. A condition is derived under which a primal-dual interior-point algorithm with a centering component is monotone. Primal-dual algorithms that are monotone in both primal and dual objective values are then proposed, and polynomial time bounds are achieved. Some arguments are also provided that show that several existing primal-dual algorithms use parameters close to one that almost always improves both objectives.

Key words. linear programming, primal-dual, interior-point methods, monotonicity

AMS subject classifications. 90C05, 65K05

1. Introduction.

Recent interest in interior-point methods for linear programming has focused on primal-dual interior-point algorithms. Numerical experiments seem to indicate that primal-dual algorithms are superior to primal-only or dual-only methods. Convergence of most of the primal-dual interior-point algorithms is based on either the decrease in the duality gap or the decrease in a potential function. Neither of these two measures tells us anything about the change in the true primal and true dual objective values. In primal-only (or dual-only) settings some work has been done to show that polynomiality and strict improvement in the objective function at each iteration can be achieved (see Anstreicher [An86], [An91], Mizuno and Nagasawa [MN92]). In this paper we address the following question: Is it possible to achieve the best polynomial bound with monotonicity in both primal and dual objective functions for primal-dual algorithms?

When one implements a primal-dual interior-point algorithm, the step sizes are often determined as a fixed ratio of the distance from the current iterate to the boundary in the primal and dual feasible regions, respectively; see [MMS89], [LMS91] for example. In that case, the next iterate may not be well defined because one of the primal and dual directions may be an unbounded direction of the corresponding feasible region. In addition, when separate step sizes are taken in the primal and dual spaces, the duality gap may increase at the next iterate. Since these phenomena do not occur if the algorithm is monotone in both objective values, the analysis of monotone algorithms is important from this practical point of view.

* Received by the editors August 3, 1992; accepted for publication (in revised form) August 9, 1993.

† Institute of Statistical Mathematics, 4–6–7 Minami-Azabu, Minato-ku, Tokyo 106, Japan. This research was performed while the author was visiting Cornell University April 1992–January 1993 as an overseas research scholar of the Ministry of Science, Education, and Culture of Japan.

‡ School of Operations Research and Industrial Engineering, Cornell University, Ithaca, New York 14853–3801. The research of this author was supported in part by the National Science Foundation, the Air Force Office of Scientific Research, and the Office of Naval Research through National Science Foundation grant DMS–8920550.

§ Department of Combinatorics and Optimization, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada. This research was performed while this author was a doctoral student at Cornell University and was supported in part by the United States Army Research Office through the Army Center of Excellence for Symbolic Methods in Algorithmic Mathematics (ACSyAM), Mathematical Sciences Institute of Cornell University contract DAAL03-91-C-0027, and also by the National Science Foundation, the Air Force Office of Scientific Research, and the Office of Naval Research through National Science Foundation grant DMS–8920550.

First, we show that the primal-dual affine-scaling algorithm of Monteiro, Adler, and Resende [MAR90] (see also [KMNY91], [MN93], [Tu92]) is (usually strictly) monotone in both primal and dual objective values. This was proved by Monteiro, Adler, and Resende [MAR90, p. 213]. Then we derive a condition on the centering parameter under which a primal-dual interior-point algorithm with a centering component is monotone in both objective values. It seems that most primal-dual interior-point algorithms (except for the affine-scaling algorithm) are not guaranteed to satisfy this condition. We show how to control the centering parameter in a way that allows monotonicity in both objectives and does not hurt the polynomial time bound proven for the nonmonotone version of a primal-dual algorithm. In particular, we study algorithms using one-sided infinity neighborhoods and two-norm neighborhoods of the central path and show that monotonicity can be achieved while keeping the iteration bounds $O(nt)$ and $O(\sqrt{n}t)$, respectively, (where t denotes the desired improvement in precision). The algorithm proposed is not only monotone, but it may also enable us to take a longer step in the sense that the centering parameter is less than in the nonmonotone version of the algorithm.

From the theoretical point of view, the complexity of primal-dual interior-point algorithms is based on a bound on the second-order term in Newton's method (see [MTY93] for example). In this paper, we obtain a new bound on the second-order term to show the polynomiality of the algorithm (see Lemma 9).

Finally, we study the anticipated critical value of the centering parameter and show that several existing primal-dual algorithms use parameters close to one that improves both objectives at almost every iteration.

2. Preliminaries. We consider linear programming problems in the following primal (P) and dual (D) forms:

$$\begin{aligned} (P) \quad \text{minimize} \quad & c^T x \\ & Ax = b, \\ & x \geq 0, \end{aligned}$$

$$\begin{aligned} (D) \quad \text{maximize} \quad & b^T y \\ & A^T y + s = c, \\ & s \geq 0, \end{aligned}$$

where $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, and $c \in \mathbb{R}^n$. We assume A has full row rank (without loss of generality) and that there exist interior solutions for both problems, i.e.,

$$\mathcal{F}_0 := \{(x, s) > 0 : x \in F(P), s \in F(D)\} \neq \emptyset,$$

where $F(P)$ and $F(D)$ denote the sets of feasible solutions for the primal and dual problems, respectively. Most of the time we deal only with s as a dual feasible solution. So, whenever we say $s \in F(D)$, we mean that $s \geq 0$ and there exists a $y \in \mathbb{R}^m$ such that $A^T y + s = c$. Given a vector denoted by a lower-case roman letter (e.g., x), the corresponding upper-case letter (e.g., X) will denote the diagonal matrix whose entries are the components of that vector (i.e., $X = \text{diag}(x)$), and e will denote the vector of ones. We denote the components of a vector using subscripts and the iterate numbers using superscripts. Whenever we ignore superscripts it will be clear from the context what the iterate number should be.

Next we describe the central path and its neighborhoods. The central path is given by the set of solutions to the following system of equations and inequalities (for

$\mu > 0$):

$$Ax = b, \qquad x \geq 0, \tag{1}$$

$$A^T y + s = c, \qquad s \geq 0, \tag{2}$$

$$Xs = \mu e. \tag{3}$$

Note that when $\mu = 0$, (1)–(3) give necessary and sufficient conditions for optimality. Analyses of the central path have been performed by several authors (see, for instance, Sonnevend [So85], Megiddo [Meg88], and Bayer and Lagarias [BL89]). Our objective is to follow this path approximately to an optimal solution. We define some neighborhoods of the central path using the notation in [MTY93]. Let $\beta \in (0,1)$ be a constant; then a 2-norm neighborhood of the central path can be defined as

$$\mathcal{N}_2(\beta) := \left\{ (x,s) \in \mathcal{F}_0 : \|Xs - \mu e\|_2 \leq \beta\mu \text{ for } \mu = \frac{x^T s}{n} \right\}.$$

Henceforth, μ always denotes $x^T s/n$. Kojima, Mizuno, and Yoshise [KMY89], Monteiro and Adler [MA89], and Mizuno, Todd, and Ye [MTY93] designed algorithms that use a 2-norm neighborhood. Using the ∞-norm or just one side of the ∞-norm, wider neighborhoods have been defined and used by Kojima, Mizuno, and Yoshise [KMY88], Mizuno, Todd, and Ye [MTY93], and Zhang and Tapia [ZT93]. Using the ∞-norm we have

$$\mathcal{N}_\infty(\beta) := \{(x,s) \in \mathcal{F}_0 : \|Xs - \mu e\|_\infty \leq \beta\mu\}$$

and using only one side of the ∞-norm,

$$\mathcal{N}_\infty^-(\beta) := \{(x,s) \in \mathcal{F}_0 : \|Xs - \mu e\|_\infty^- \leq \beta\mu\}.$$

Here, for $u \in \mathbb{R}^n$, $\|u\|_\infty^- := -\min\{0, \min\{u_j\}\}$. Clearly, for a given $\beta \in (0,1)$, $\mathcal{N}_2(\beta)$ is the smallest and $\mathcal{N}_\infty^-(\beta)$ the largest of the three neighborhoods defined here.

Suppose we have an interior-point solution $(x,s) \in \mathcal{F}_0$. Then a search direction (dx, ds) can be generated by solving the following set of equalities (see Kojima, Mizuno, and Yoshise [KMY88]):

$$Adx = 0, \tag{4}$$

$$A^T dy + ds = 0, \tag{5}$$

$$Sdx + Xds = \gamma\mu e - Xs, \tag{6}$$

where $\gamma \in [0,1]$ is a constant. The solution of (4)–(6) is

$$dx(\gamma) := -X^{1/2}S^{-1/2}P_{\bar{A}}(X^{1/2}S^{1/2} - \gamma\mu X^{-1/2}S^{-1/2})e,$$
$$ds(\gamma) := -X^{-1/2}S^{1/2}(I - P_{\bar{A}})(X^{1/2}S^{1/2} - \gamma\mu X^{-1/2}S^{-1/2})e,$$

where $\bar{A} := AX^{1/2}S^{-1/2}$, and $P_{\bar{A}} := I - \bar{A}^T(\bar{A}\bar{A}^T)^{-1}\bar{A}$, the projection matrix into the null space of $\bar{A}$. The next iterates in terms of α can be written as

$$x(\alpha) = x + \alpha dx(\gamma),$$
$$s(\alpha) = s + \alpha ds(\gamma).$$

A primal-dual interior-point algorithm is one that, given an iterate $(x,s) \in \mathcal{F}_0$, defines the next iterate as $(x(\alpha), s(\alpha))$ derived as above for some $\gamma \in [0,1]$ and some

$\alpha > 0$ such that $(x(\alpha), s(\alpha)) \in \mathcal{F}_0$ (the centering parameter γ and the step size α may vary from one iteration to the next). We define

$$v := \frac{1}{\sqrt{\mu}} X^{1/2} S^{1/2} e,$$
$$w := \sqrt{\mu} X^{-1/2} S^{-1/2} e.$$

Note that $\|v\|_2^2 = n$ and $v^T w = n$. Also observe that (x, s) is on the central path if and only if $v = w = e$. We use v_p to denote the projection of v onto the null space of $\bar{A}$, etc., and v_q to denote the projection of v onto the image space of $\bar{A}^T$, etc. Then we have that

$$\begin{aligned} Xs - \mu e &= \mu V(v - w), \\ dx(\gamma) &= -\sqrt{\mu} X^{1/2} S^{-1/2}(v_p - \gamma w_p), \\ ds(\gamma) &= -\sqrt{\mu} X^{-1/2} S^{1/2}(v_q - \gamma w_q), \end{aligned} \tag{7}$$

where $V = \operatorname{diag}(v)$. From these equalities, we see that

$$\mathcal{N}_2(\beta) = \{(x, s) \in \mathcal{F}_0 : \|V(v - w)\|_2 \le \beta\}, \tag{8}$$

$$\mathcal{N}_\infty^-(\beta) = \{(x, s) \in \mathcal{F}_0 : \|V(v - w)\|_\infty^- \le \beta\}, \tag{9}$$

$$dX(\gamma) ds(\gamma) = \mu(V_p - \gamma W_p)(v_q - \gamma w_q), \tag{10}$$

where $dX(\gamma) = \operatorname{diag}(dx(\gamma))$, $V_p = \operatorname{diag}(v_p)$, and $W_p = \operatorname{diag}(w_p)$.

3. Monotonicity. First we show that the primal-dual affine-scaling algorithms (see Monteiro, Adler, and Resende [MAR90], Kojima et al. [KMNY91], Mizuno and Nagasawa [MN93], and Tunçel [Tu92]) are (usually strictly) monotone in both objectives. As mentioned in the introduction, this was essentially proved in [MAR90].

Note that for any $(x, s) \in \mathcal{F}_0$, we have $X^{1/2} S^{-1/2} A^T y + X^{1/2} S^{1/2} e = X^{1/2} S^{-1/2} c$ or

$$\bar{A}^T y + \sqrt{\mu} v = \bar{c},$$

where $\bar{c} := X^{1/2} S^{-1/2} c$. Projecting the vectors on both sides of the equality onto the null space of $\bar{A}$, we see that

$$v_p = \frac{1}{\sqrt{\mu}} \bar{c}_p. \tag{11}$$

An affine-scaling algorithm results from setting $\gamma = 0$ at each iteration. Then we get

$$\begin{aligned} c^T x(\alpha) - c^T x &= \alpha c^T dx(0) \\ &= -\alpha\sqrt{\mu}(X^{1/2} S^{-1/2} c)^T v_p \\ &= -\alpha\sqrt{\mu}\bar{c}^T v_p \\ &= -\alpha\mu\|v_p\|_2^2. \end{aligned} \tag{12}$$

For each $s(\alpha)$, there is a $y(\alpha)$ such that $(y(\alpha), s(\alpha))$ is feasible for (D). In the same way, we get

$$\begin{aligned} b^T y(\alpha) - b^T y &= (Ax)^T y(\alpha) - (Ax)^T y \\ &= -x^T s(\alpha) + x^T s \\ &= -\alpha x^T ds(0) \\ &= \alpha\mu\|v_q\|_2^2. \end{aligned} \tag{13}$$

We immediately have the following theorem.

THEOREM 1. *A primal-dual affine-scaling algorithm is strictly monotone in both problems unless all the primal or all the dual solutions are optimal.*

Proof. Since a primal-dual affine-scaling algorithm is monotone in both objectives by (12) and (13), we only prove strict monotonicity.

Suppose $\|v_p\|_2 = 0$. Then $\bar{c}_p = 0$ by (11). So $\hat{s} := 0$ is feasible in (D); therefore, for any $x \in F(P)$, x is optimal in (P) and $\hat{s}$ in (D) by complementary slackness.

Now suppose $\|v_q\|_2 = 0$. Then $b = Ax = \sqrt{\mu}\bar{A}v = 0$, which implies $\hat{x} := 0$ is feasible in (P). Hence, for any s that is feasible in (D), s is optimal in (D) and $\hat{x}$ is optimal in (P) by complementary slackness. □

From now on, without loss of generality, we assume that

$$0 \;<\; \|v_p\|_2^2 \;<\; n.$$

In the same way as we obtained (12), we have

$$\begin{aligned} c^T x(\alpha) - c^T x &= -\alpha\sqrt{\mu}(X^{1/2}S^{-1/2}c)^T(v_p - \gamma w_p) \\ &= -\alpha\mu v_p^T(v_p - \gamma w_p). \end{aligned}$$

So we have the following identities for improvements in the objective function values:

$$c^T x(\alpha) - c^T x = -\alpha\mu\left(\|v_p\|_2^2 - \gamma v_p^T w_p\right) \tag{14}$$

and, similarly,

$$b^T y(\alpha) - b^T y = \alpha\mu\left(\|v_q\|_2^2 - \gamma v_q^T w_q\right). \tag{15}$$

We define γ_P and γ_D as *constant-cost centering values* for the primal and dual problems respectively so that $c^T x(\alpha) = c^T x$ and $b^T y(\alpha) = b^T y$:

$$\gamma_P := \frac{\|v_p\|_2^2}{v_p^T w_p}, \qquad \gamma_D := \frac{\|v_q\|_2^2}{v_q^T w_q}, \tag{16}$$

where we define $\gamma_P := +\infty$ or $\gamma_D := +\infty$ when the corresponding denominator is zero. Note that if (x, s) is on the central path $(v = w)$ then $\gamma_P = \gamma_D = 1$. The next lemma describes a relation between γ_P and γ_D.

LEMMA 2. *Either* $\gamma_P = \gamma_D = 1$ *or*

$$\gamma_P \in (0,1) \text{ and } \gamma_D \in (-\infty, 0) \cup (1, \infty]$$

or

$$\gamma_D \in (0,1) \text{ and } \gamma_P \in (-\infty, 0) \cup (1, \infty]$$

Proof. We see that $n = v^T v = \|v_p\|_2^2 + \|v_q\|_2^2$ and $n = v^T w = v_p^T w_p + v_q^T w_q$. If $v_p^T w_p = \|v_p\|_2^2$ then $v_q^T w_q = \|v_q\|_2^2$, so we have $\gamma_P = \gamma_D = 1$. If $v_p^T w_p > \|v_p\|_2^2$ then $\gamma_P \in (0,1)$ and $\gamma_D \in (-\infty, 0) \cup (1, \infty]$. Otherwise, $\gamma_D \in (0,1)$ and $\gamma_P \in (-\infty, 0) \cup (1, \infty]$. □

Define

$$\bar{\gamma} := \gamma_P \text{ if } \gamma_P \in (0,1], \qquad \text{and } \bar{\gamma} := \gamma_D \text{ otherwise .}$$

LEMMA 3. *If $0 \le \gamma \le \bar{\gamma}$ $(0 \le \gamma < \bar{\gamma})$, then the primal objective function is nonincreasing (decreasing) along the direction $dx(\gamma)$ and the dual objective function is nondecreasing (increasing) along the direction $ds(\gamma)$.*

Proof. The proof follows from Lemma 2 and (14) and (15). □

Now we have a sufficient condition under which a primal-dual interior-point algorithm is monotone in both objective values.

THEOREM 4. *An algorithm whose search directions are $dx(\gamma)$ and $ds(\gamma)$ for $0 \le \gamma \le \bar{\gamma}$ $(0 \le \gamma < \bar{\gamma})$ at each iterate is (strictly) monotone in both primal and dual objective values.*

Proof. The proof directly follows from Lemma 3. □

If the exceptional case ($\|v_p\|_2^2 = 0$ or $\|v_p\|_2^2 = n$) is excluded, Theorem 1 is a special case of Theorem 4. Although the condition in Theorem 4 is simple, none of the primal-dual interior-point algorithms, except for the affine-scaling algorithm, seems to use values for the centering parameter that are guaranteed to satisfy it. Here we describe a strictly monotone algorithm, which is a simple variant of a primal-dual path-following algorithm [KMY88], [KMY89], [MA89], [MTY93]. Let $\mathcal{N}$ be a neighborhood of the central path and $\gamma' \in (0,1)$ be a constant. Suppose $(x^0, s^0) \in \mathcal{N}$ with $(x^0)^T s^0 \le 2^t$ is given.

ALGORITHM
$k := 0$
While $((x^k)^T s^k > 2^{-t})$ do
  $(x, s) := (x^k, s^k)$,
  compute v_p, v_q, w_p, w_q, γ_P and γ_D,
  $\bar{\gamma} := \gamma_P$ if $\gamma_P \in (0,1]$ and $\bar{\gamma} := \gamma_D$ otherwise,
  $\gamma := \gamma' \bar{\gamma}$,
  choose the maximum step size $\alpha \in (0,1)$ such that $(x + \alpha dx(\gamma), s + \alpha ds(\gamma)) \in \mathcal{N}$,
  $(x^{k+1}, s^{k+1}) := (x + \alpha dx(\gamma), s + \alpha ds(\gamma))$,
  $k := k + 1$
end

If we use $\gamma := \gamma'$ in the algorithm above, it becomes a well-known path-following algorithm. In that case, the algorithm terminates in $O(nt)$ iterations when $\mathcal{N} = \mathcal{N}_\infty^-(\beta)$ and $\gamma' \in (0,1)$ is a constant, and it terminates in $O(\sqrt{n}t)$ iterations when $\mathcal{N} = \mathcal{N}_2(\beta)$ and $\gamma' = 1 - \delta/\sqrt{n}$ for a positive constant δ (for suitable values of the constants see [KMY88], [KMY89], [MA89], [MTY93]). Taking $\mathcal{N} := \mathcal{N}_\infty^-(\beta)$ and $\gamma' \in (0,1)$ in the algorithm above defines Algorithm I. Algorithm II will be defined by setting $\mathcal{N} := \mathcal{N}_2(\beta)$ and $\gamma' := 1 - \delta/\sqrt{n}$. In §4, we show that Algorithm I terminates in $O(nt)$ iterations and Algorithm II terminates in $O(\sqrt{n}t)$ iterations.

Note that if $\bar{\gamma}$ is small, the centering parameter γ in the algorithm is small even if γ' is close to 1. This observation distinguishes Algorithms I and II from the path-following algorithms in [KMY88], [KMY89], [MA89], [MTY93].

4. Analysis. The following results (Lemmas 5 and 6) are standard in the primal-dual framework (see for instance Mizuno, Todd, and Ye [MTY93]).

LEMMA 5. (a) $x_j(\alpha)s_j(\alpha) = \mu[(1-\alpha)v_j^2 + \alpha\gamma] + \alpha^2 dx_j(\gamma)ds_j(\gamma)$.

(b) $\mu(\alpha) := x(\alpha)^T s(\alpha)/n = (1 - \alpha + \alpha\gamma)\mu$.

Proof. Both (a) and (b) directly follow from the definitions and it holds that $dx(\gamma)^T ds(\gamma) = 0$. □

Part (b) of this lemma shows how the duality gap varies with the step size α. We would like $\alpha(1-\gamma)$ to be large. The next result shows how large α can be.

LEMMA 6. (a) $(x,s) \in \mathcal{N}_\infty^-(\beta)$ *and*

$$\alpha \leq \min\left\{\frac{\beta\gamma\mu}{\|dX(\gamma)ds(\gamma)\|_\infty^-}, 1\right\}$$

implies $(x(\alpha), s(\alpha)) \in \mathcal{N}_\infty^-(\beta)$.

(b) $(x,s) \in \mathcal{N}_2(\beta)$ *and*

$$\alpha \leq \min\left\{\frac{\beta\gamma\mu}{\|dX(\gamma)ds(\gamma)\|_2}, 1\right\}$$

implies $(x(\alpha), s(\alpha)) \in \mathcal{N}_2(\beta)$.

Proof. (a) Suppose $(x,s) \in \mathcal{N}_\infty^-(\beta)$ and

$$\alpha \leq \min\left\{\frac{\beta\gamma\mu}{\|dX(\gamma)ds(\gamma)\|_\infty^-}, 1\right\}.$$

By (9), $\|V(v-w)\|_\infty^- \leq \beta$. By using $Vw = e$, we see that

$$v_j^2 \geq 1-\beta \quad \text{for each} \quad j. \tag{17}$$

So, using Lemma 5 and the bound on α, we get (for all j)

$$\begin{aligned} x_j(\alpha)s_j(\alpha) &\geq \mu[(1-\alpha)(1-\beta)+\alpha\gamma] - \alpha\beta\gamma\mu \\ &= \mu(1-\alpha+\alpha\gamma)(1-\beta) \\ &= \mu(\alpha)(1-\beta). \end{aligned}$$

Since $\alpha \in [0,1]$ and $\gamma > 0$, the right-hand side is always positive. So $(x(\alpha), s(\alpha)) \in \mathcal{F}_0$. Hence $(x(\alpha), s(\alpha)) \in \mathcal{N}_\infty^-(\beta)$.

(b) Suppose $(x,s) \in \mathcal{N}_2(\beta)$ and

$$\alpha \leq \min\left\{\frac{\beta\gamma\mu}{\|dX(\gamma)ds(\gamma)\|_2}, 1\right\}.$$

Then the feasibility of $(x(\alpha), s(\alpha))$ can be verified as in (a). Using (6), the bound on α, and Lemma 5, we obtain

$$\begin{aligned} \|X(\alpha)s(\alpha) - \mu(\alpha)e\|_2 &= \|(1-\alpha)Xs + \alpha\gamma\mu e + \alpha^2 dX(\gamma)ds(\gamma) - (1-\alpha+\alpha\gamma)\mu e\|_2 \\ &= \|(1-\alpha)(Xs-\mu e) + \alpha^2 dX(\gamma)ds(\gamma)\|_2 \\ &\leq (1-\alpha)\|Xs-\mu e\|_2 + \alpha\beta\gamma\mu \\ &\leq (1-\alpha)\beta\mu + \alpha\beta\gamma\mu \\ &= \beta\mu(\alpha). \qquad \square \end{aligned}$$

One way of showing a bound on the number of iterations while keeping the algorithm monotone is to show that there exists an $\epsilon > 0$ such that $\bar{\gamma} \geq \epsilon$ for all iterates. It seems hard to find such an ϵ that is close to one (or even bounded away from zero). Instead, we show that the second-order terms $\|dXds\|_\infty^-$ and $\|dXds\|_2$ can be bounded by a multiple of $\bar{\gamma}\mu$. This is a new way of estimating the second-order terms. Since the bound we show is a multiple of $\bar{\gamma}$, it implies that whenever $\bar{\gamma}$ is small, the second-order terms are small, hence allowing us to take a larger step and still stay in

the desired neighborhood. Since $dX(\gamma)ds(\gamma) = \mu(V_p - \gamma W_p)(v_q - \gamma w_q)$ by (10), we get bounds on $\|v_p - \gamma w_p\|_2$ and $\|v_q - \gamma w_q\|_2$ in the following two lemmas.

LEMMA 7. *If $\gamma_P < +\infty$, then $\|v_p - \gamma_P w_p\|_2 \le |\gamma_P| \|v_p - w_p\|_2$; if $\gamma_D < +\infty$, then $\|v_q - \gamma_D w_q\|_2 \le |\gamma_D| \|v_q - w_q\|_2$.*

Proof. Note that $v_p^T(v_p - \gamma_P w_p) = 0$ from the definition (16) of γ_P. So,

$$\begin{aligned}\|v_p - \gamma_P w_p\|_2^2 &= (v_p - \gamma_P w_p)^T (v_p - \gamma_P w_p) \\ &= (\gamma_P v_p - \gamma_P w_p)^T (v_p - \gamma_P w_p) \\ &\le |\gamma_P| \|v_p - w_p\|_2 \|v_p - \gamma_P w_p\|_2.\end{aligned}$$

If $\|v_p - \gamma_P w_p\|_2 = 0$ the first inequality holds; otherwise, we divide both sides by $\|v_p - \gamma_P w_p\|_2$ to get the result. In the same way, we are able to prove the second inequality. □

LEMMA 8. (a) *Let $\gamma \in (0, 1]$. If $(x, s) \in \mathcal{N}_\infty^-(\beta)$ then*

$$\|v - \gamma w\|_2 \le \sqrt{\frac{2n}{1-\beta}}.$$

(b) *Let $\gamma \in [1 - \delta/\sqrt{n}, 1]$. If $(x, s) \in \mathcal{N}_2(\beta)$ then*

$$\|v - \gamma w\|_2 \le \frac{\beta + \delta}{\sqrt{1-\beta}} \quad \text{and} \quad \|v - w\|_2 \le \frac{\beta}{\sqrt{1-\beta}}.$$

Proof. (a) If $(x, s) \in \mathcal{N}_\infty^-(\beta)$, we have (17). Using (17) and $Vw = e$, we get

$$w_j^2 \le 1/(1-\beta) \quad \text{for each } j. \tag{18}$$

Hence $\|v - \gamma w\|_2^2 = v^T v - 2\gamma v^T w + \gamma^2 w^T w \le n + w^T w \le 2n/(1-\beta)$.

(b) Since $(x, s) \in \mathcal{N}_2(\beta) \subset \mathcal{N}_\infty^-(\beta)$, we have (17) and $\|V(v - w)\|_2 \le \beta$ (see (8)). So,

$$\begin{aligned}\|v - \gamma w\|_2 &\le (1-\gamma)\|v\|_2 + \gamma \|v - w\|_2 \\ &\le (1-\gamma)\sqrt{n} + \gamma \|V(v-w)\|_2 \|V^{-1}e\|_\infty \\ &\le \delta + \gamma\beta/\sqrt{1-\beta} \\ &\le (\delta + \beta)/\sqrt{1-\beta}.\end{aligned}$$

If $\gamma = 1$, we can set $\delta = 0$ to get the last inequality. □

LEMMA 9. *Let $\beta \in (0, 1)$ and $\delta > 0$ be constants, and let $\bar{\gamma} = \gamma_P$ if $\gamma_P \in (0, 1]$ and $\bar{\gamma} = \gamma_D$ otherwise.*

(a) *If $(x, s) \in \mathcal{N}_\infty^-(\beta)$, $\gamma' \in (0, 1]$, and $\gamma = \gamma'\bar{\gamma}$, then we have*

$$\|dX(\gamma)ds(\gamma)\|_\infty^- \le 4\bar{\gamma} n \mu/(1-\beta).$$

(b) *If $(x, s) \in \mathcal{N}_2(\beta)$, $\gamma' = 1 - \delta/\sqrt{n}$, and $\gamma = \gamma'\bar{\gamma}$, then we have*

$$\|dX(\gamma)ds(\gamma)\|_2 \le \bar{\gamma}\mu(3 + \delta)^2/(1-\beta).$$

Proof. Without loss of generality, we assume that $\gamma_P \in (0, 1]$ and $\bar{\gamma} = \gamma_P$ in the analysis below.

(a) Let $(x, s) \in \mathcal{N}_\infty^-(\beta)$ and $\gamma' \in (0, 1]$. Since $\gamma = \gamma'\gamma_P$, we have

$$\|v_p - \gamma w_p\|_2 \le \|v_p - \gamma_P w_p\|_2 + \gamma_P(1 - \gamma')\|w_p\|_2. \tag{19}$$

Using Lemma 7, Lemma 8(a), and (18), we get

$$\begin{aligned}\|v_p - \gamma w_p\|_2 &\le \gamma_P \sqrt{\frac{2n}{1-\beta}} + \gamma_P(1-\gamma')\sqrt{\frac{n}{1-\beta}} \\ &\le 2\gamma_P \sqrt{\frac{2n}{1-\beta}}.\end{aligned}$$

Using Lemma 8(a) we also have

$$\|v_q - \gamma w_q\|_2 \le \sqrt{\frac{2n}{1-\beta}}.$$

By (10), we obtain

$$\begin{aligned}\|dX(\gamma)ds(\gamma)\|_\infty^- &= \mu\|(V_p - \gamma W_p)(v_q - \gamma w_q)\|_\infty^- \\ &\le \mu\|v_p - \gamma w_p\|_2 \|v_q - \gamma w_q\|_2 \\ &\le 4\gamma_P n\mu/(1-\beta).\end{aligned}$$

(b) Let $(x,s) \in \mathcal{N}_2(\beta)$ and $\gamma' = 1 - \delta/\sqrt{n}$. Using (18), (19), and Lemma 7, we have

$$\begin{aligned}\|v_p - \gamma w_p\|_2 &\le \gamma_P \|v - w\|_2 + \gamma_P(1-\gamma')\sqrt{\frac{n}{1-\beta}} \\ &\le \gamma_P \left(\frac{\beta}{\sqrt{1-\beta}} + \frac{\delta}{\sqrt{1-\beta}}\right).\end{aligned}$$

The last inequality follows from Lemma 8(b). We also see

$$\begin{aligned}\|v_q - \gamma w_q\|_\infty &= \|(v - \gamma w) - (v_p - \gamma w_p)\|_\infty \\ &\le \|v\|_\infty + \|w\|_\infty + \|v_p - \gamma w_p\|_2 \\ &\le \sqrt{1+\beta} + \frac{1}{\sqrt{1-\beta}} + \frac{\beta+\delta}{\sqrt{1-\beta}} \\ &\le \frac{(3+\delta)}{\sqrt{1-\beta}}.\end{aligned}$$

So

$$\begin{aligned}\|dX(\gamma)ds(\gamma)\|_2 &= \mu\|(V_q - \gamma W_q)(v_p - \gamma w_p)\|_2 \\ &\le \mu\|v_q - \gamma w_q\|_\infty \|v_p - \gamma w_p\|_2 \\ &\le \gamma_P \mu(3+\delta)^2/(1-\beta). \qquad \square\end{aligned}$$

THEOREM 10. *Let $\beta \in (0,1)$ and $\delta > 0$ be constants independent of the input data of (P) and (D).*

(a) *Algorithm* I *($\mathcal{N} = \mathcal{N}_\infty^-(\beta)$ and $\gamma' \in (0,1)$ is independent of the input data) terminates in $O(nt)$ iterations and is strictly monotone in both objectives.*

(b) *Algorithm* II *($\mathcal{N} = \mathcal{N}_2(\beta)$ and $\gamma' = 1 - \delta/\sqrt{n}$) terminates in $O(\sqrt{n}t)$ iterations and is strictly monotone in both objectives.*

Proof. The strict monotonicity follows from Theorem 4. (a) Since $(x,s) \in \mathcal{N}_\infty^-(\beta)$ throughout Algorithm I, by Lemma 6(a), Lemma 9(a), and $\gamma = \gamma'\bar{\gamma}$, we get

$$(x(\alpha), s(\alpha)) \in \mathcal{N}_\infty^-(\beta) \quad \text{for} \quad \alpha \le \frac{\beta(1-\beta)\gamma'}{4n}.$$

Then by Lemma 5(b), we have

$$\begin{aligned}(x^{k+1})^T s^{k+1} &\le (1-(1-\gamma)\beta(1-\beta)\gamma'/4n)(x^k)^T s^k \\ &\le (1-(1-\gamma')\beta(1-\beta)\gamma'/4n)(x^k)^T s^k,\end{aligned}$$

which implies $(x^k)^T z^k \le 2^{-t}$ for $k = O(nt)$.

(b) As in the proof of (a), the result follows from Lemma 5(b), Lemma 6(b), Lemma 9(b), and $\gamma = \gamma'\bar{\gamma}$. In particular, since $(x,s) \in \mathcal{N}_2(\beta)$ throughout Algorithm II and $\gamma = \gamma'\bar{\gamma}$, we find

$$(x(\alpha), s(\alpha)) \in \mathcal{N}_2(\beta) \quad \text{for} \quad \alpha \le \frac{\beta(1-\beta)\gamma'}{(3+\delta)^2}. \qquad \square$$

5. Typical values of γ_P and γ_D. Consider Algorithm II. The proof of Theorem 10(b) above shows that we can choose α at least equal to some constant (independent of n) value $\hat{\alpha} \in (0,1]$ when $(x,s) \in \mathcal{N}_2(\beta)$, $\gamma' = 1 - \delta/\sqrt{n}$ and $\gamma = \gamma'\bar{\gamma}$. Hence, whenever $\bar{\gamma} \le \hat{\gamma}$ for some constant $\hat{\gamma} \in (0,1)$, we could achieve a constant factor reduction in the duality gap by Lemma 5. In this section we consider whether or not this is likely to occur.

Note that in Algorithm I, we always have $\gamma \le \hat{\gamma}$ for constant $\hat{\gamma} \in (0,1)$ (in fact, $\hat{\gamma} = \gamma'$), but we can only guarantee that α is at least some constant divided by n by Lemmas 6 and 9; hence typical values of γ_P and γ_D are of less interest.

For definiteness, choose $\beta = \frac{1}{3}$, and suppose $(x,s) \in \mathcal{N}_2(\beta)$. By (8), we have

$$\|Vv - e\|_2 \le \beta,$$

and so

$$\|v - e\|_2 \le \beta \tag{20}$$

since each v_j is at least as close to one as is v_j^2. In addition, $w - e = W(e - v)$, so

$$\begin{aligned}\|w - e\|_2 &\le \|w\|_\infty \|v - e\|_2 \\ &\le \frac{\beta}{\sqrt{1-\beta}}\end{aligned} \tag{21}$$

using (18). Since $\beta = \frac{1}{3}$, we get

$$\|v - e\|_2 \le \frac{1}{2}, \qquad \|w - e\|_2 \le \frac{1}{2}, \tag{22}$$

and from $\|e\|_2 = \sqrt{n}$ we deduce

$$\theta(v,e) \le \arcsin\left(\frac{1}{2\sqrt{n}}\right), \qquad \theta(w,e) \le \arcsin\left(\frac{1}{2\sqrt{n}}\right), \tag{23}$$

where $\theta(t,u)$ denotes the angle between vectors t and u.

Now γ_P and γ_D are related to the angles between the projections of v and w into the null space U of $\bar{A}$ or its orthogonal complement. If $v \neq w$, it is possible to choose U (or $U^\perp$) so that $v_p^T w_p$ (or $v_q^T w_q$) is negative. However, since from (23)

$$\theta(v, w) \leq \arcsin\left(\frac{1}{\sqrt{n}}\right),$$

it appears that for "most" subspaces U, $v_p^T w_p$ will be close to $\|v_p\|_2^2$ and hence γ_P close to one. Here we would like to make this statement, in some sense, precise. As in [MTY93], we assume that

(24) U is a random subspace of $\mathbb{R}^n$ of dimension $d := n - m$, drawn from the unique distribution on such subspaces that is invariant under orthogonal transformations.

To analyze this situation, we proceed as in the proof of Lemma 6 of [MTY93]. The orthogonal invariance of the subspace U implies that we can instead assume that U is the fixed subspace

$$\tilde{U} := \{x \in \mathbb{R}^n : x_{d+1} = \cdots = x_n = 0\},$$

and that e, v, and w are replaced by $\tilde{e}$, $\tilde{v}$, and $\tilde{w}$, where these vectors are the results of applying the orthogonal transformation ϑ taking U to $\tilde{U}$ to the vectors e, v, and w. Hence

(25) $\tilde{e}$ is uniformly distributed on the unit sphere of radius $\sqrt{n}$ in $\mathbb{R}^n$,

and

$$\|\tilde{v} - \tilde{e}\|_2 \leq \frac{1}{2}, \qquad \|\tilde{w} - \tilde{e}\|_2 \leq \frac{1}{2}, \tag{26}$$

$$\theta(\tilde{v}, \tilde{e}) \leq \arcsin\left(\frac{1}{2\sqrt{n}}\right), \qquad \theta(\tilde{w}, \tilde{e}) \leq \arcsin\left(\frac{1}{2\sqrt{n}}\right). \tag{27}$$

Let $\tilde{e}_p$, $\tilde{v}_p$, and $\tilde{w}_p$ denote the projections of $\tilde{e}$, $\tilde{v}$, and $\tilde{w}$ onto $\tilde{U}$; they are also the images under ϑ of e_p, v_p, and w_p. Hence

$$\gamma_P = \frac{\|\tilde{v}_p\|_2^2}{\tilde{v}_p^T \tilde{w}_p}. \tag{28}$$

Now $\|\tilde{v}_p - \tilde{e}_p\|_2 \leq \frac{1}{2}$ from (26), so that

$$\|\tilde{v}_p\|_2 \in \left[\|\tilde{e}_p\|_2 - \frac{1}{2}, \|\tilde{e}_p\|_2 + \frac{1}{2}\right], \tag{29}$$

and similarly,

$$\|\tilde{w}_p\|_2 \in \left[\|\tilde{e}_p\|_2 - \frac{1}{2}, \|\tilde{e}_p\|_2 + \frac{1}{2}\right]. \tag{30}$$

Also, $\tilde{v}_p^T \tilde{w}_p = \frac{1}{2}\left(\|\tilde{v}_p\|_2^2 + \|\tilde{w}_p\|_2^2 - \|\tilde{v}_p - \tilde{w}_p\|_2^2\right)$, so since $\|\tilde{v}_p - \tilde{w}_p\|_2 \leq \|\tilde{v} - \tilde{w}\|_2 \leq 1$,

$$\tilde{v}_p^T \tilde{w}_p \in \left[\left(\|\tilde{e}_p\|_2 - \frac{1}{2}\right)^2 - \frac{1}{2}, \left(\|\tilde{e}_p\|_2 + \frac{1}{2}\right)^2\right]. \tag{31}$$

Hence, if $\|\tilde{e}_p\|_2 \geq \frac{3}{2}$,

$$\gamma_P \in \left[\left(\frac{\|\tilde{e}_p\|_2 - \frac{1}{2}}{\|\tilde{e}_p\|_2 + \frac{1}{2}} \right)^2, \frac{\left(\|\tilde{e}_p\|_2 + \frac{1}{2}\right)^2}{\left(\|\tilde{e}_p\|_2 - \frac{1}{2}\right)^2 - \frac{1}{2}} \right], \tag{32}$$

$$\gamma_P \geq 1 - \frac{2}{\|\tilde{e}_p\|_2 + \frac{1}{2}}. \tag{33}$$

THEOREM 11. *Suppose assumption* (24) *holds and* $d = \Omega(n)$. *Then, with probability approaching* 1 *as* $n \to \infty$,

$$\gamma_P \geq 1 - \frac{\xi}{\sqrt{n}}$$

for some constant ξ.

Proof. We use (33). As in Lemma 6 of [MTY93], we can suppose

$$\tilde{e} = \sqrt{n} \left(\frac{\lambda_1}{\|\lambda\|_2}, \ldots, \frac{\lambda_n}{\|\lambda\|_2} \right)^T$$

from (25), where $\lambda_1, \ldots, \lambda_n$ are independent normal random variables with mean zero and variance one. Then

$$\tilde{e}_p = \sqrt{n} \left(\frac{\lambda_1}{\|\lambda\|_2}, \ldots, \frac{\lambda_d}{\|\lambda\|_2}, 0, \ldots, 0 \right)^T$$

and

$$\|\tilde{e}_p\|_2^2 = n \left(\frac{\lambda_1^2 + \cdots + \lambda_d^2}{\lambda_1^2 + \cdots + \lambda_n^2} \right).$$

The quantity in parentheses is a beta random variable with parameters $d/2$ and $m/2$, which has mean d/n and variance $2dm/n^2(n+2) \geq 1/(2n+4)$. Using Chebyshev's inequality,

$$\|\tilde{e}_p\|_2^2 \geq n \left(\frac{d}{2n} \right)$$

with probability approaching one as $n \to \infty$. Since $d = \Omega(n)$, this shows $\|\tilde{e}_p\|_2 = \Omega(\sqrt{n})$ with high probability, and hence completes the proof. □

Since $U^\perp$ satisfies (24) if U does, the same result shows that $\gamma_D \geq 1 - \xi/\sqrt{n}$ with high probability as long as $m = \Omega(n)$. These results suggest that we cannot expect to find $\bar{\gamma}$ ($= \gamma'\gamma_P$ or $\gamma'\gamma_D$) smaller than $1 - \kappa/\sqrt{n}$ for some constant κ very often. Note that several primal-dual interior-point algorithms use a value for γ that is $1 - \zeta/\sqrt{n}$ for some constant ζ (e.g., [KMY89], [MA89]); our result indicates that this value is close to one that improves both objectives at almost every iteration.

6. Concluding remarks. In an implementation of the strictly monotone algorithm described in §3, we may use a small parameter value of γ' (for example, $\gamma' = 1/n$) and long step sizes such that the next primal and dual iterates are located a fixed ratio (say .99) of the way from the current point to the boundary of the corresponding feasible region. In that case, the next iterate is well defined and the duality gap decreases strictly, although these phenomena have not been shown in nonmonotone versions of the algorithm.

REFERENCES

[An86] K. M. ANSTREICHER, *A monotonic projective algorithm for fractional linear programming*, Algorithmica, 1 (1986), pp. 483–498.

[An91] ———, *On monotonicity in the scaled potential algorithm for linear programming*, Linear Algebra Appl., 152 (1991), pp. 223–232.

[BL89] D. A. BAYER AND J. C. LAGARIAS, *The nonlinear geometry of linear programming:* I. *Affine and projective scaling trajectories*, II. *Legendre transform coordinates and central trajectories*, Trans. Amer. Math. Soc., 314 (1989), pp. 499–581.

[KMY88] M. KOJIMA, S. MIZUNO, AND A. YOSHISE, *A primal-dual interior point algorithm for linear programming*, in Progress in Mathematical Programming, Interior Point and Related Methods, N. Megiddo, ed., Springer-Verlag, New York, 1988, pp. 29–47.

[KMY89] ———, *A polynomial time algorithm for a class of linear complementarity problems*, Math. Programming, 44 (1989), pp. 1–26.

[KMNY91] M. KOJIMA, N. MEGIDDO, T. NOMA, AND A. YOSHISE, *A unified approach to interior point algorithms for linear complementarity problems*, Lecture Notes in Computer Science 538, Springer-Verlag, New York, 1991.

[LMS91] I. J. LUSTIG, R. E. MARSTEN, AND D. F. SHANNO, *Computation experience with a primal-dual interior point method for linear programming*, Linear Algebra Appl., 152 (1991), pp. 191–222.

[MMS89] K. A. MCSHANE, C. L. MONMA, AND D. SHANNO, *An implementation of a primal-dual interior point method for linear programming*, ORSA J. Comp., 1 (1989), pp. 70–83.

[Meg88] N. MEGIDDO, *Pathways to the optimal set in linear programming*, in Progress in Mathematical Programming, Interior Point and Related Methods, N. Megiddo, ed., Springer-Verlag, New York, 1988, pp. 131–158.

[MN92] S. MIZUNO AND A. NAGASAWA, *Strict monotonicity in Todd's low-complexity algorithm for linear programming*, Oper. Res. Lett., 12 (1992), pp. 59–64.

[MN93] ———, *A primal-dual affine scaling potential reduction algorithm for linear programming*, Math. Programming, 63 (1993), pp. 119–131.

[MTY93] S. MIZUNO, M. J. TODD, AND Y. YE, *On adaptive-step primal-dual interior-point algorithms for linear programming*, Math. Oper. Res., 18 (1993), pp. 964–981.

[MA89] R. C. MONTEIRO AND I. ADLER, *Interior path following primal-dual algorithms, Part* I: *Linear programming*, Math. Programming, 44 (1989), pp. 27–42.

[MAR90] R. C. MONTEIRO, I. ADLER, AND M. G. C. RESENDE, *A polynomial time primal-dual affine scaling algorithm for linear and convex quadratic programming and its power series extension*, Math. Oper. Res., 15 (1990), pp. 191–214.

[So85] G. SONNEVEND, *An analytical center for polyhedrons and new classes of global algorithms for linear (smooth convex) programming*, in Lecture Notes in Control and Information Sciences 84, Springer-Verlag, New York, 1985, pp. 866–876.

[Tu92] L. TUNÇEL, *A Note on the Primal-Dual Affine Scaling Algorithms*, TR-1004 (1992), School of Operations Research and Industrial Engineering, Cornell University, Ithaca, NY.

[ZT93] Y. ZHANG AND R. A. TAPIA, *A quadratically convergent polynomial primal-dual interior point algorithm for linear programming*, SIAM J. Optim., 3 (1993), pp. 118–133.

SIAM J. OPTIMIZATION
Vol. 4, No. 3, pp. 626–636, August 1994
© 1994 Society for Industrial and Applied Mathematics
011

THE ITERATED KALMAN SMOOTHER AS A GAUSS–NEWTON METHOD*

BRADLEY M. BELL†

Abstract. The Kalman smoother is known to be the maximum likelihood estimator when the measurement and transition functions are affine; i.e., a linear function plus a constant. A new proof of this result is presented that shows that the Kalman smoother decomposes a large least squares problem into a sequence of much smaller problems. The iterated Kalman smoother is then presented and shown to be a Gauss–Newton method for maximizing the likelihood function in the nonaffine case. This method takes advantage of the decomposition obtained with the Kalman smoother.

Key words. nonlinear Kalman smoothing, maximum likelihood, nonlinear least squares, Gauss–Newton, nonlinear recursive least squares

AMS subject classification. 93E11

1. Introduction.

In this paper we present Kalman smoothing as a large unconstrained least squares problem. We derive the Kalman smoother as a decomposition method for solving the least squares problem when the measurement and transition functions are affine. (In the literature, the affine case is often referred to as the linear case.) This leads directly to the iterated Kalman smoother, which we show is a Gauss–Newton method for solving the least squares problem in the nonaffine case.

Iterating the Kalman filter can greatly improve results when the measurement or transition functions are not affine [3], [8], [14]. Algorithms that iterate the Kalman smoother are a current topic of interest (for example, [13]). It is important that these algorithms use a method that has a convergence theory. The Gauss–Newton method has such a theory associated with it (see [6, §10.2]).

The Kalman smoother is known to be the maximum likelihood estimator when the measurement and transition functions are affine [11]. Most research into the nonaffine case has been concentrated on determining the expected value of the state vector given the available data. This is equivalent to determining the minimum variance estimator [5]. Implementable methods for calculating the minimum variance estimator have been extended to the case where the measurement and transition function are quadratic [10].

We show that, when the measurement and transition functions are affine, the Kalman smoother decomposes the maximum likelihood problem into a sequence of much smaller problems. This view leads directly to the iterated smoother that takes advantage of this decomposition in the nonaffine case. We expect many other results to follow from this view of the Kalman smoothers and filters.

In §2, we define the optimization problem that corresponds to Kalman smoothing. In §3, we present the Kalman filtering theorem that enables us to decompose the optimization problem in the affine case. In §4, we present the Kalman smoother and show that it solves the affine case. In §5, we present the iterated Kalman smoother and show that it is a Gauss–Newton method for solving the nonaffine case.

* Received by the editors August 17, 1992; accepted for publication (in revised form) August 17, 1993.

† Applied Physics Laboratory, College of Ocean and Fishery Sciences, University of Washington, 1013 N.E. 40th Street, Seattle, Washington 98105 (brad@apl.washington.edu). This research was supported by Office of Naval Research contract N00014-87-K-0760.

2. Problem definition. Our goal is to estimate the state sequence $\{x_k : k = 1, \ldots, N\}$. Let n be the dimension of the state space; i.e., $x_k \in \mathbb{R}^n$ for each k. We are given $\{z_k : k = 1, \ldots, N\}$, a sequence of vector valued measurements, and $\hat{x}_1$, an estimate of the initial state. Let $m(k)$ be the dimension of the data space for the kth time point. (This dimension need not be the same from one time point to the next.)

The data values are related to the state values by the measurement equation

$$z_k = h_k(x_k) + v_k,$$

where $h_k : \mathbb{R}^n \to \mathbb{R}^{m(k)}$ represents a noiseless measurement as a function of the state and $v_k \in \mathbb{R}^{m(k)}$ is normally distributed with mean zero and variance R_k. We assume the functions $\{h_k\}$ and matrices $\{R_k\}$ are known.

The state values at each time point are related to the state values at the previous time point by the transition equation

$$x_{k+1} = g_k(x_k) + w_k,$$

where $g_k : \mathbb{R}^n \to \mathbb{R}^n$ represents a noiseless transition and $w_k \in \mathbb{R}^n$ is normally distributed with mean zero and variance Q_k. We assume the functions $\{g_k\}$ and matrices $\{Q_k\}$ are known.

The initial state estimate $\hat{x}_1$ is related to the initial state by

$$x_1 = \hat{x}_1 + e_1,$$

where e_1 is normally distributed with mean zero and variance P_1. We assume the matrix P_1 is known.

Assumption. The random variables $\{v_k : k = 1, \ldots, N\}$, $\{w_k : k = 1, \ldots, N-1\}$, and e_1 are independent normally distributed with mean zero and known variances $\{R_k : k = 1, \ldots, N\}$, $\{Q_k : k = 1, \ldots, N-1\}$, and P_1, respectively, where each of the variance matrices is positive definite.

The likelihood function for our problem is given in [2]. Maximizing the likelihood function is equivalent to minimizing

$$\begin{aligned} L(x_1, x_2, \ldots, x_N) = \Bigg\{ & (x_1 - \hat{x}_1)^T P_1^{-1} (x_1 - \hat{x}_1) \\ & + \sum_{k=1}^{N} (z_k - h_k(x_k))^T R_k^{-1} (z_k - h_k(x_k)) \\ & + \sum_{k=1}^{N-1} (x_{k+1} - g_k(x_k))^T Q_k^{-1} (x_{k+1} - g_k(x_k)) \Bigg\}. \end{aligned} \tag{1}$$

Note that the function L is known because all the quantities on the right-hand side are given except the state values $\{x_k : k = 1, \ldots, N\}$.

The affine smoother and iterated smoother solve the problem

$$\text{minimize } L(x_1, x_2, \ldots, x_N) \quad \text{with respect to} \quad (x_1, x_2, \ldots, x_N)$$

for the affine and nonaffine cases, respectively.

3. The Kalman filter theorem. If the functions $\{h_k : k = 1, \ldots, N\}$ and $\{g_k : k = 1, \ldots, N-1\}$ are affine, the problem of minimizing L with respect to $(x_1, \ldots, x_N)$ can be decomposed into a sequence of minimization problems with respect to x_k for $k = 1, \ldots, N$. This decomposition is justified by Theorem 3.4 which is presented in this section.

If a function $h(x)$ is known to be affine, its gradient is constant and is denoted by ∇h instead of $\nabla h(x)$ (see Definition 4.1.7 of [6] for a definition of the gradient). This notation is used throughout the rest of this paper.

Lemma 3.1 is a derivative identity. If two expressions have the same derivatives, they can replace each other as part of an objective function without changing the solution. This result is usually stated in terms of the solutions of certain least squares problems [4].

LEMMA 3.1 (Recursive least squares). *Suppose the function $h : \mathbb{R}^n \to \mathbb{R}^m$ is affine, the matrices P and R are symmetric positive definite, $\hat{x} \in \mathbb{R}^n$, and $z \in \mathbb{R}^m$.*

$$\begin{array}{ll} \textit{If} & f(x) = (x-\hat{x})^T P^{-1}(x-\hat{x}) + (z-h(x))^T R^{-1}(z-h(x)), \\ \textit{then} & \nabla f(x) = \nabla_x[(x-u)^T U^{-1}(x-u)], \\ \textit{where} & U = [P^{-1} + \nabla h R^{-1} \nabla h^T]^{-1} \quad \textit{and} \quad u = \hat{x} + U \nabla h R^{-1}(z - h(\hat{x})). \end{array}$$

Proof. The gradient of f is given by

$$\nabla f(x) = 2P^{-1}(x-\hat{x}) - 2\nabla h R^{-1}(z - h(x)).$$

Because h is affine, we can substitute $(\nabla h^T)(x-\hat{x}) + h(\hat{x})$ for $h(x)$ to obtain

$$\begin{aligned} \nabla f(x) &= 2[P^{-1} + \nabla h R^{-1} \nabla h^T](x-\hat{x}) - 2\nabla h R^{-1}(z - h(\hat{x})), \\ &= 2U^{-1}(x-\hat{x}) - 2\nabla h R^{-1}(z - h(\hat{x})). \end{aligned}$$

It follows that $\nabla f(u)$ is zero and $\nabla^2 f(x)$ is constant and equal to $2U^{-1}$. Thus

$$\nabla f(x) = \nabla f(u) + \nabla^2 f(u)(x-u) = 2U^{-1}(x-u) = \nabla_x[(x-u)^T U^{-1}(x-u)] ,$$

which is what we needed to prove. □

The following observation and lemma are used in the proof of Lemma 3.3. The observation can be proven by solving for the value of w at which the gradient of the objective function is zero and substituting that value back into the objective function. A proof of Lemma 3.2 can be found in [1].

Observation. If A is symmetric positive definite, then

$$\min_w \quad w^T A w - 2b^T w = -b^T A^{-1} b .$$

LEMMA 3.2 (Matrix inversion). *Suppose that n and m are positive integers and $Q \in \mathbb{R}^{n \times n}$, $H \in \mathbb{R}^{m \times n}$, and $U \in \mathbb{R}^{m \times m}$, where Q and U are positive definite. It follows that*

$$(Q + H^T U H)^{-1} = Q^{-1} - Q^{-1} H^T [U^{-1} + H Q^{-1} H^T]^{-1} H Q^{-1}.$$

LEMMA 3.3 (Least squares projection). *Suppose the function $g : \mathbb{R}^n \to \mathbb{R}^n$ is affine, the matrices U and Q are symmetric positive definite, and $u \in \mathbb{R}^n$.*

$$\begin{array}{ll} \textit{If} & f(x,y) = (x-u)^T U^{-1}(x-u) + (y-g(x))^T Q^{-1}(y-g(x)), \\ \textit{and} & q(y) = \min\{f(x,y) : x \in \mathbb{R}^n\}, \\ \textit{then} & q(y) = (y-\hat{y})^T F^{-1}(y-\hat{y}), \\ \textit{where} & F = [Q + \nabla g^T U \nabla g] \quad \textit{and} \quad \hat{y} = g(u). \end{array}$$

Proof. The lemma defines $\hat{y}$ to be $g(u)$ and assumes that g is affine. Substituting $\hat{y} + (\nabla g^T)(x-u)$ for $g(x)$ in the definition for f and regrouping terms, we obtain

$$f(x,y) = \{\, (x-u)^T[U^{-1} + \nabla g Q^{-1} \nabla g^T](x-u) \\ - 2(y-\hat{y})^T Q^{-1}(\nabla g^T)(x-u) \,+\, (y-\hat{y})^T Q^{-1}(y-\hat{y}) \,\}.$$

We used the fact that $(y-\hat{y})^T Q^{-1}(\nabla g^T)(x-u)$ is a scalar and equal to its transpose to obtain the equality above. We now apply the observation above with

$$A = [U^{-1} + \nabla g Q^{-1} \nabla g^T], \quad b = \nabla g Q^{-1}(y-\hat{y}), \quad w = (x-u)$$

to conclude that

$$\begin{aligned} q(y) &= -b^T A^{-1} b + (y-\hat{y})^T Q^{-1}(y-\hat{y}), \\ &= \{\, -(y-\hat{y})^T Q^{-1} \nabla g^T [U^{-1} + \nabla g Q^{-1} \nabla g^T]^{-1} \nabla g Q^{-1}(y-\hat{y}) \\ &\qquad + (y-\hat{y})^T Q^{-1}(y-\hat{y}) \,\}\,, \\ &= (y-\hat{y})^T B (y-\hat{y}), \end{aligned}$$

where

$$B = Q^{-1} - Q^{-1} \nabla g^T [U^{-1} + \nabla g Q^{-1} \nabla g^T]^{-1} \nabla g Q^{-1}\,.$$

Applying Lemma 3.2 with H equal to ∇g we obtain

$$B = (Q + \nabla g^T U \nabla g)^{-1} = F^{-1},$$

which completes the proof of the lemma. □

The Kalman filter only estimates the most recent state, i.e., x_N. Theorem 3.4 can be used to derive the Kalman filter as the value of x_N that corresponds to a minimum of L (see Lemma 4.1).

THEOREM 3.4 (Kalman filter). *Suppose the functions $h : \mathbb{R}^n \to \mathbb{R}^m$ and $g : \mathbb{R}^n \to \mathbb{R}^n$ are affine, the function $f(y,w)$ is convex, the matrices P, Q, and R are symmetric positive definite, $\hat{x} \in \mathbb{R}^n$, and $z \in \mathbb{R}^m$. Define the problem*

$$\underset{(x,y,w)}{\text{minimize}} \{\, (x-\hat{x})^T P^{-1}(x-\hat{x}) \,+\, (z-h(x))^T R^{-1}(z-h(x)) \\ + (y-g(x))^T Q^{-1}(y-g(x)) \,+\, f(y,w) \,\}. \tag{2}$$

There is an $\bar{x}$ such that the triple $(\bar{x},\bar{y},\bar{w})$ solves the problem above if and only if the pair $(\bar{y},\bar{w})$ solves the problem

$$\underset{(y,w)}{\text{minimize}}\ (y-\hat{y})^T F^{-1}(y-\hat{y}) \,+\, f(y,w),$$

where F and $\hat{y}$ are defined by

$$\begin{aligned} U &= [P^{-1} + \nabla h R^{-1} \nabla h^T]^{-1}, & F &= Q + \nabla g^T U \nabla g, \\ u &= \hat{x} + U \nabla h R^{-1}(z - h(\hat{x})), & \hat{y} &= g(u). \end{aligned}$$

Proof. There is an $\bar{x}$ such that the triple $(\bar{x},\bar{y},\bar{w})$ solves the problem in (2) if and only if the pair $(\bar{y},\bar{w})$ solves the problem

$$\underset{(y,w)}{\text{minimize}} \{\, f(y,w) + \min_x [(x-\hat{x})^T P^{-1}(x-\hat{x}) + (z-h(x))^T R^{-1}(z-h(x)) \\ + (y-g(x))^T Q^{-1}(y-g(x))\,]\}.$$

TABLE 1
Affine smoother input.

Name	Description
N	Number of time points
$\hat{x}_1$	Estimate of initial state value
P_1	Initial state estimate covariance
z_k	Measurement value for $k = 1, \ldots, N$
h_k	Affine measurement function for $k = 1, \ldots, N$
g_k	Affine transition function for $k = 1, \ldots, N-1$
R_k	Measurement covariance for $k = 1, \ldots, N$
Q_k	Transition covariance for $k = 1, \ldots, N-1$

TABLE 2
Affine smoother definitions.

Name	Description
u_k	The corrected state estimate for $k = 1, \ldots, N$
U_k	Covariance of the corrected state estimate for $k = 1, \ldots, N$
$\hat{x}_k$	The predicted state estimate for $k = 2, \ldots, N$
P_k	Covariance of the predicted state estimate for $k = 2, \ldots, N$
B_k	for $k = 1, \ldots, N-1$
s_k	The smoothed state estimate for $k = 1, \ldots, N$

Note that the "min" in the expression above must be attained and is unique because P, Q, and R are positive definite. We can apply Lemma 3.1 and the convexity of f to conclude that a pair $(\bar{y}, \bar{w})$ solves the problem above if and only if it solves

$$\underset{(y,w)}{\text{minimize}} \{ f(y,w) + \min_x [(x-u)^T U^{-1}(x-u) + (y-g(x))^T Q^{-1}(y-g(x))] \}.$$

We can apply Lemma 3.3 to conclude that a pair $(\bar{y}, \bar{w})$ solves the problem above if and only if it solves

$$\underset{(y,w)}{\text{minimize}} \, (y-\hat{y})^T F^{-1}(y-\hat{y}) + f(y,w) .$$

This completes the proof of the theorem. □

4. The affine smoother. In this section we present the affine smoother and show that it minimizes the function L defined in (1). Throughout this section we assume that the functions $\{h_k : k = 1, \ldots, N\}$ and $\{g_k : k = 1, \ldots, N-1\}$ are affine.

The affine smoother is defined in Fig. 1. We call the assignments in this algorithm definitions because once a value is assigned it is never changed. This is not the most efficient way to write the algorithm but it makes the proofs in this section much clearer. The values and functions in Table 1 are known input to the affine smoother. The values in Table 2 are defined by the affine smoother. The functions $\{h_k : k = 1, \ldots, N\}$ and $\{g_k : k = 1, \ldots, N-1\}$ are listed as parameters to the affine smoother because they are replaced by other functions when the iterated smoother uses the affine smoother (see Fig. 2). The result, namely, $(s_1, \ldots, s_N)$ is the value returned by the affine smoother.

The first part of the affine smoother, which generates the sequence $\{u_k : k = 1, \ldots, N\}$, is the alternative form of the Kalman filter [12], [1]. The standard form of the Kalman filter can be obtained using Lemma 3.2 to transform the expression defining U_k in the algorithm.

affine smoother ($h_1, \ldots, h_N, g_1, \ldots, g_{N-1}$)
  for $k = 1$ to N begin
    $U_k = [P_k^{-1} + \nabla h_k R_k^{-1} \nabla h_k^T]^{-1}$
    $u_k = \hat{x}_k + U_k \nabla h_k R_k^{-1}(z_k - h_k(\hat{x}_k))$
    if $k \neq N$ begin
      $P_{k+1} = Q_k + \nabla g_k^T U_k \nabla g_k$
      $\hat{x}_{k+1} = g_k(u_k)$
    end
  end
  $s_N = u_N$
  for $k = N - 1$ to 1 begin
    $B_k = [P_k^{-1} + \nabla h_k R_k^{-1} \nabla h_k^T + \nabla g_k Q_k^{-1} \nabla g_k^T]$
    $s_k = \hat{x}_k + B_k^{-1}[\nabla h_k R_k^{-1}(z_k - h_k(\hat{x}_k)) + \nabla g_k Q_k^{-1}(s_{k+1} - g_k(\hat{x}_k))]$
  end
  return ($s_1, \ldots, s_N$)
end

FIG. 1. *Affine smoother algorithm.*

for $i = 1$ to $M - 1$ begin
  for $k = 1$ to N begin
    function $h_k^i(x) = h_k(s_k^i) + \nabla h_k(s_k^i)^T(x - s_k^i)$
    function $g_k^i(x) = g_k(s_k^i) + \nabla g_k(s_k^i)^T(x - s_k^i)$
  end
  $(s_1^{i+1}, \ldots, s_N^{i+1})$ = affine smoother $(h_1^i, \ldots, h_N^i, g_1^i, \ldots, g_{N-1}^i)$
end

FIG. 2. *Iterated smoother algorithm.*

The matrix P_1 is given, and the affine smoother defines the matrices $\{P_k : k = 2, \ldots, N\}$. Define the sequence of functions $\{L_j : j = 1, \ldots, N\}$ by

$$L_j(x_j, x_{j+1}, \ldots, x_N) = \left\{ \begin{aligned} &(x_j - \hat{x}_j)^T P_j^{-1}(x_j - \hat{x}_j) \\ &+ \sum_{k=j}^{N} (z_k - h_k(x_k))^T R_k^{-1}(z_k - h_k(x_k)) \\ &+ \sum_{k=j}^{N-1} (x_{k+1} - g_k(x_k))^T Q_k^{-1}(x_{k+1} - g_k(x_k)) \end{aligned} \right\}.$$

LEMMA 4.1. *If the functions* $\{h_k : k = 1, \ldots, N\}$ *and* $\{g_k : k = 1, \ldots, N-1\}$ *are affine, a vector* $(\bar{x}_j, \ldots, \bar{x}_N)$ *minimizes* L_j *if and only if there exists a sequence* $\{\bar{x}_k : k = 1, \ldots, j-1\}$ *such that* $(\bar{x}_1, \ldots, \bar{x}_N)$ *minimizes* L.

Proof. The lemma is true for $j = 1$ because L_1 is equal to L. Suppose by induction that the lemma is true for $j = k$, where $k < N$. We complete the proof by showing it is true for $j = k + 1$.

Define the function f by

$$f(x_{k+1},\ldots,x_N) = L_{k+1}(x_{k+1},\ldots,x_N) - (x_{k+1}-\hat{x}_{k+1})^T P_{k+1}^{-1}(x_{k+1}-\hat{x}_{k+1})\,.$$

It follows from the definition of L_k, L_{k+1}, and f that

$$\begin{aligned} L_k(x_k,\ldots,x_N) = \{\, & f(x_{k+1},\ldots,x_N) \\ & + (x_k-\hat{x}_k)^T P_k^{-1}(x_k-\hat{x}_k) \\ & + (z_k-h_k(x_k))^T R_k^{-1}(z_k-h_k(x_k)) \\ & + (x_{k+1}-g_k(x_k))^T Q_k^{-1}(x_{k+1}-g_k(x_k))\}\,. \end{aligned}$$

We now apply Theorem 3.4 with $x = x_k$, $y = x_{k+1}$, and $w = (x_{k+2},\ldots,x_N)$, together with the definitions of P_{k+1} and $\hat{x}_{k+1}$ in the algorithm, to conclude that $(\bar{x}_{k+1},\ldots,\bar{x}_N)$ minimizes

$$f(x_{k+1},\ldots,x_N) + (x_{k+1}-\hat{x}_{k+1})^T P_{k+1}^{-1}(x_{k+1}-\hat{x}_{k+1}) = L_{k+1}(x_{k+1},\ldots,x_N)$$

if and only if there exists an $\bar{x}_k$ such that $(\bar{x}_k,\ldots,\bar{x}_N)$ minimizes L_k. This completes the induction for $j = k+1$ that completes the proof. □

THEOREM 4.2. *If the functions* $\{h_k : k = 1,\ldots,N\}$ *and* $\{g_k : k = 1,\ldots,N-1\}$ *are affine, the vector* $(s_1,\ldots,s_N)$*, which is returned by the affine smoother, minimizes the function* L *defined in* (1).

Proof. We prove this theorem by showing that for each j the vector $(s_j,\ldots,s_N)$ minimizes L_j. The case $j = 1$ corresponds to the theorem. This statement is proven by induction on j starting with $j = N$.

We begin with the case $j = N$. By Lemma 4.1 it suffices to show that s_N minimizes

$$L_N(x_N) = (x_N-\hat{x}_N)^T P_N^{-1}(x_N-\hat{x}_N) + (z_N-h_N(x_N))^T R_N^{-1}(z_N-h_N(x_N)).$$

The argmin of L_N can be calculated by solving for the zero of its gradient; i.e., L_N is minimized by the x_N such that

$$0 = P_N^{-1}(x_N-\hat{x}_N) - \nabla h_N R_N^{-1}(z_N-h_N(x_N)).$$

Because h_N is affine, we can substitute $h_N(\hat{x}_N) + (\nabla h_N^T)(x_N-\hat{x}_N)$ for $h_N(x_N)$. Solving for $x_N-\hat{x}_N$, we obtain

$$x_N-\hat{x}_N = [P_N^{-1} + \nabla h_N R_N^{-1}\nabla h_N^T]^{-1}\nabla h_N R_N^{-1}(z_N-h_N(\hat{x}_N))\,. \tag{3}$$

The affine smoother defines s_N as equal to u_N, where

$$\begin{aligned} U_N &= [P_N^{-1} + \nabla h_N R_N^{-1}\nabla h_N^T]^{-1}, \\ u_N &= \hat{x}_N + U_N \nabla h_N R_N^{-1}(z_N-h_N(\hat{x}_N)). \end{aligned}$$

Substituting u_N for x_N satisfies (3), hence s_N minimizes L_N. This completes the case $j = N$.

Suppose that the inductive hypothesis is true for $j = k+1$. We complete the proof of the theorem by showing that the inductive hypothesis is true for $j = k$.

By induction $(s_{k+1},\ldots,s_N)$ minimizes L_{k+1}. By Lemma 4.1 there exists $\{\bar{x}_i : i = 1,\ldots,k\}$ such that

$$(\bar{x}_1,\ldots,\bar{x}_k,s_{k+1},\ldots,s_N)$$

TABLE 3
Iterated smoother input used by affine smoother.

Name	Description
N	Number of time points
$\hat{x}_1$	Estimate of initial state value
P_1	Initial state estimate covariance
z_k	Measurement value for $k = 1, \ldots, N$
R_k	Measurement covariance for $k = 1, \ldots, N$
Q_k	Transition covariance for $k = 1, \ldots, N-1$

TABLE 4
Other iterated smoother input.

M	Number of iterations of the Gauss–Newton method to use
h_k	Measurement function for $k = 1, \ldots, N$
g_k	Transition function for $k = 1, \ldots, N-1$
s_k^1	$(s_1^1, \ldots, s_N^1)$ is initial approximate minimum of L

minimizes L. By Lemma 4.1 $(\bar{x}_k, s_{k+1}, \ldots, s_N)$ minimizes L_k. So it suffices to show that s_k minimizes $L_k(x_k, s_{k+1}, \ldots, s_N)$ with respect to x_k. Taking the gradient of L_k with respect to x_k at $(x_k, s_{k+1}, \ldots, s_N)$ and setting it equal to zero, we have

$$0 = P_k^{-1}(x_k - \hat{x}_k) - \nabla h_k R_k^{-1}(z_k - h_k(x_k)) - \nabla g_k Q_k^{-1}(s_{k+1} - g_k(x_k)).$$

Substituting $h_k(\hat{x}_k) + (\nabla h_k^T)(x_k - \hat{x}_k)$ for $h_k(x_k)$, $g_k(\hat{x}_k) + (\nabla g_k^T)(x_k - \hat{x}_k)$ for $g_k(x_k)$, and using the definition of B_k in the algorithm, we obtain

$$\nabla h_k R_k^{-1}(z_k - h_k(\hat{x}_k)) + \nabla g_k Q_k^{-1}(s_{k+1} - g_k(\hat{x}_k)) = B_k(x_k - \hat{x}_k).$$

The affine smoother defines s_k such that it satisfies this equation for x_k (when $k < N$). This shows that the gradient of $L_k(x_k, s_{k+1}, \ldots, s_N)$ with respect to x_k is zero at s_k and thereby completes the proof of the theorem. □

5. The iterated smoother. In this section we present the iterated Kalman smoother and show that it is a Gauss–Newton method for minimizing the function L, defined in (1).

The iterated smoother is defined in Fig. 2. We call the assignments in this algorithm definitions because, once a value is assigned, it is never changed. This is not the most efficient way to write the algorithm, but it makes the proof of Theorem 5.1 much clearer. The values in Table 3 are known input to the iterated smoother that are used by the affine smoother. Table 4 contains the other input to the iterated smoother. The values and functions in Table 5 are defined by the iterated smoother. The iterated smoother passes the affine approximations $\{h_k^i\}$ and $\{g_k^i\}$ as input to the affine smoother. The corresponding affine smoother results are stored in $\{s_k^i\}$. The vector $(s_1^M, \ldots, s_N^M)$ is the final result of the iterated smoother.

We choose the number of iterations M as our stopping criterion for the iterated smoother. Other stopping criteria may be more efficient. For example, the change in the state values between iterations or the change in the objective function may be a better stopping criterion.

THEOREM 5.1. *The iterated smoother is a Gauss–Newton method for minimizing the function L defined in* (1).

TABLE 5
Iterated smoother definitions.

Name	Description
h_k^i	Affine approximation for h_k for $k = 1, \ldots, N$ and $i = 1, \ldots, M-1$
g_k^i	Affine approximation for g_k for $k = 1, \ldots, N$ and $i = 1, \ldots, M-1$
s_k^i	$(s_1^i, \ldots, s_N^i)$ is an approximate minimum of L for $i = 2, \ldots, M$

Proof. Given a sequence of square matrices $\{T_j : j = 1, \ldots, J\}$ we use the notation

$$[T_1, \ldots, T_J] = \begin{pmatrix} T_1 & 0 & \ldots & 0 \\ 0 & T_2 & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \ldots & T_J \end{pmatrix}.$$

Define the matrix

$$A = [P_1^{-1}, R_1^{-1}, Q_1^{-1}, \ldots, Q_{N-1}^{-1}, R_N^{-1}].$$

Define the function

$$v(x_1, \ldots, x_N) = \begin{pmatrix} x_1 - \hat{x}_1 \\ z_1 - h_1(x_1) \\ x_2 - g_1(x_1) \\ \vdots \\ x_N - g_{N-1}(x_{N-1}) \\ z_N - h_N(x_N) \end{pmatrix}.$$

Using these definitions the objective function can be expressed as

$$L(x_1, \ldots, x_N) = v(x_1, \ldots, x_N)^T A v(x_1, \ldots, x_N).$$

The matrices P_1, $\{R_k : k = 1, \ldots, N\}$ and $\{Q_k : k = 1, \ldots, N-1\}$ are positive definite. Let the matrix C_1 be a square root of P_1^{-1}; C_{2k} is a square root of R_k^{-1} for $k = 1, \ldots, N$; and C_{2k+1} is a square root of Q_k^{-1} for $k = 1, \ldots, N-1$. (If B is a positive definite matrix, then B^{-1} is positive definite and B has a square root S such that $B = S^T S$.) Define the matrix C by

$$C = [C_1, C_2, \ldots, C_{2N}].$$

We can express the objective function as

$$\begin{aligned} L(x_1, \ldots, x_N) &= v(x_1, \ldots, x_N)^T C^T C v(x_1, \ldots, x_N) \\ &= \|Cv(x_1, \ldots, x_N)\|^2. \end{aligned}$$

The Gauss–Newton method is an algorithm for minimizing functions of the form $\|r(y)\|^2$, where $r : \mathbb{R}^p \to \mathbb{R}^q$ is twice continuously differentiable and $\|\cdot\|$ is the Euclidean norm on $\mathbb{R}^q$. Define $r(y) = Cv(y)$ so that $L(y) = \|r(y)\|^2$. If y^i is the ith iterate of the Gauss–Newton method, the $(i+1)$th iterate (see [6, §10.2]) is the solution to the problem

$$\underset{y}{\text{minimize}} \, \|r(y^i) + \nabla r(y^i)^T (y - y^i)\|^2.$$

Replacing r by its definition, we have the equivalent problem

$$\underset{y}{\text{minimize}} \, \|C[v(y^i) + \nabla v(y^i)^T (y - y^i)]\|^2 . \tag{4}$$

Thus it suffices to show that if y^i equals $(s_1^i, \ldots, s_N^i)$, then $(s_1^{i+1}, \ldots, s_N^{i+1})$ solves the problem above. Substituting $(s_1^i, \ldots, s_N^i)$ for y^i and $(x_1, \ldots, x_N)$ for y, we have

$$v(y^i) + \nabla v(y^i)^T (y - y^i) = \begin{pmatrix} s_1^i - \hat{x}_1 & + & x_1 - s_1^i \\ z_1 - h_1(s_1^i) & - & \nabla h_1(s_1^i)(x_1 - s_1^i) \\ s_2^i - g_1(s_1^i) & + & x_2 - s_2^i - \nabla g_1(s_1^i)(x_1 - s_1^i) \\ & \vdots & \\ z_N - h_N^i(s_N^i) & - & \nabla h_N(s_N^i)(x_N - s_N^i) \end{pmatrix}$$

$$v(y^i) + \nabla v(y^i)^T (y - y^i) = \begin{pmatrix} x_1 - \hat{x}_1 \\ z_1 - h_1^i(x_1) \\ x_2 - g_1^i(x_1) \\ \vdots \\ z_N - h_N^i(x_N) \end{pmatrix},$$

where $\{h_k^i\}$ and $\{g_k^i\}$ are the approximations defined in the iterated Kalman smoother. Thus the objective function in (4) is equal to

$$L^i(x_1, x_2, \ldots, x_N) = \Big\{ \|C_1(x_1 - \hat{x}_1)\|^2 + \sum_{k=1}^{N} \|C_{2k}(z_k - h_k^i(x_k))\|^2 + \sum_{k=1}^{N-1} \|C_{2k+1}(x_{k+1} - g_k^i(x_k))\|^2 \Big\},$$

where $(x_1, \ldots, x_N) = y$.

From the definitions of C_k, we obtain the following equivalent representation for the objective function in (4):

$$\Big\{ (x_1 - \hat{x}_1)^T P_1^{-1} (x_1 - \hat{x}_1) + \sum_{k=1}^{N} (z_k - h_k^i(x_k))^T R_k^{-1} (z_k - h_k^i(x_k)) + \sum_{k=1}^{N-1} (x_{k+1} - g_k^i(x_k))^T Q_k^{-1} (x_{k+1} - g_k^i(x_k)) \Big\}.$$

We know by Theorem 4.2 that $(s_1^{i+1}, \ldots, s_N^{i+1})$ minimizes this function which completes the proof of Theorem 5.1. □

6. Conclusions and suggestions. Kalman smoothing can be considered a decomposition method for a large least squares problem. We have presented a new derivation of the affine smoother from this point of view. The affine Kalman filter can also be derived along the same lines (using Lemma 4.1). We have also extended the smoother to the more general case using the Gauss–Newton algorithm. We expect this view to generate many new algorithms and results for nonlinear Kalman smoothing. For example, it may be useful to combine the iterated smoother with a

line search along the lines suggested for the Gauss–Newton method [7]. It may also be useful to apply the Gauss–Newton method to the continuous time case. (See [9] for a discussion of nonlinear continuous time filtering.) It would also be interesting to investigate the convergence of the iterated Kalman smoother in terms of its input values.

Acknowledgments. The author thanks Dr. Frederick Cathey and Dr. Robert Odom for helpful discussions. He also thanks Drs. Bruce Howe and James Mercer for supporting this research.

REFERENCES

[1] B. D. O. ANDERSON AND J. B. MOORE, *Optimal Filtering*, Prentice-Hall, Englewood Cliffs, NJ, 1987, pp. 138–140.

[2] T. W. ANDERSON, *An Introduction to Multivariate Statistical Analysis*, John Wiley and Sons, New York, 1958, Section 8.2.1.

[3] B. M. BELL AND F. W. CATHEY, *The iterated Kalman filter update as a Gauss–Newton method*, IEEE Trans. Automatic Control, 38 (1993), pp. 294–297.

[4] W. L. BROGAN, *Modern Control Theory*, Quantum Publishers, New York, 1974, pp. 102–103.

[5] D. E. CATLIN, *Estimation, Control, and the Discrete Kalman Filter*, Springer-Verlag, New York, 1989, p. 68.

[6] J. E. DENNIS AND R. B. SCHNABEL, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, Englewood Cliffs, NJ, 1983.

[7] R. FLETCHER, *Practical Methods of Optimization*, 2nd ed., John Wiley, New York, 1987, p. 114.

[8] A. GELB, ED., *Applied Optimal Estimation*, MIT Press, Cambridge, MA, 1974, p. 190.

[9] G. JUMARIE, *Nonlinear filtering: A weighted mean squares approach and a Bayesian one via the maximum entropy principle*, Signal Processing, 21 (1990), pp. 323–338.

[10] D. F. LIANG, *Comparisons of nonlinear filters for systems with non-negligible nonlinearities*, in Advances in the Techniques and Technology of the Application of Nonlinear Filters and Kalman Filters, C. T. Leondes, ed., AGARDograph No. 256, North Atlantic Treaty Organization Advisory Group for Aerospace Research and Development, 1982, p. 16-17.

[11] H. E. RAUCH, F. TUNG, AND C. T. STRIEBEL, Section 3.2 *Maximum likelihood estimates of linear dynamic systems*, AIAA J., 3 (1965), §3.2, pp. 1447–1448.

[12] H. W. SORENSON, *Kalman filtering techniques*, in Kalman Filtering Theory and Applications, H. W. Sorenson, ed., IEEE Press, New York, 1966, pp. 107–108.

[13] L. TAN, J. DOWLING, C. MCCORKELL, AND H. MCCABE, *State estimation for optimal control of a non-linear system*, in Proc. IECON '91, IEEE CH22976-91-0000-2235, 1991, pp. 2235–2239.

[14] R. P. WISHNER, J. A. TABACZYNSKI, AND M. ATHANS, *A comparison of three non-linear filters*, Automatica, 5 (1969), pp. 487–496.

SIAM J. OPTIMIZATION
Vol. 4, No. 3, pp. 637–648, August 1994
© 1994 Society for Industrial and Applied Mathematics
012

ONE-PARAMETRIC SEMI-INFINITE OPTIMIZATION: ON THE STABILITY OF THE FEASIBLE SET*

HUBERTUS TH. JONGEN,† JAN-J. RÜCKMANN,‡ AND GERD-WILHELM WEBER†

Abstract. This paper studies a global stability property of the (noncompact) feasible set $M(H,G,t)$ of a semi-infinite optimization problem defined by finitely many equations $H(x,t)=0$ and, perhaps, infinitely many inequalities $G(x,t,y)\leq 0$ that depend on a real parameter t that varies in a compact parameter interval T.

Global stability refers to the homeomorphy of $M[H,G,t_1]$ and $M[H,G,t_2]$ for any parameter values $t_1,t_2\in T$.

It is shown that the overall validity of a so-called extended Mangasarian–Fromovitz constraint qualification at infinity (a constraint qualification taking parameter information into account) is sufficient for the stability mentioned.

Key words. one-parametric semi-infinite optimization, stability of the feasible set, homeomorphy, extended Mangasarian–Fromovitz constraint qualification at infinity

AMS subject classifications. 90C30, 90C31

1. Introduction and main result.

Let R^n be the n-dimensional Euclidean space. If $U\subset R^n$ is open, we denote the space of k-times continuously differentiable mappings from U to R^m by $C^k(U,R^m)$, $k\geq 1$. If $f\in C^1(R^n,R)$ then $Df(x)$ represents the derivative (row vector) of f at $x\in R^n$.

In this paper we consider the (perhaps noncompact) feasible set of a semi-infinite optimization problem depending on a real parameter t and given in the following standard way:

$$M[H,G,T]=\left\{ (x,t)\in R^n\times T \;\middle|\; \begin{array}{l} h_i(x,t)=0,\ i\in I \\ G(x,t,y)\leq 0,\ y\in Y\subset R^r \end{array} \right\},$$

where $h_i\in C^1(R^n\times R,R)$, $i\in I=\{1,\dots,m\}$, $m<n$, $G\in C^1(R^n\times R\times R^r,R)$, $H=(h_1,\dots,h_m)$ and $T\subset R$ is a compact (parameter) interval. Furthermore, the set $Y\subset R^r$ is assumed to be compact and for $\bar t\in T$ we define

$$M[H,G,\bar t\,]=\{x\in R^n | h_i(x,\bar t\,)=0,\ i\in I,\ G(x,\bar t,y)\leq 0,\ y\in Y\}.$$

The following constraint qualification (first used in [9]) will play an essential role in this paper.

The extended Mangasarian–Fromovitz constraint qualification (EMFCQ) is said to hold at $\bar x\in M[H,G,\bar t\,]$ if

(i) $\{D_x h_i(\bar x,\bar t\,),\ i\in I\}$ is a linearly independent set and (ii) there exists a vector $\xi\in R^n$ satisfying

$$\begin{aligned} D_x h_i(\bar x,\bar t\,)\xi &= 0, \qquad i\in I,\\ D_x G(\bar x,\bar t,y)\xi &< 0, \qquad y\in E_G(\bar x,\bar t\,), \end{aligned}$$

where $E_G(\bar x,\bar t\,)=\{y\in Y | G(\bar x,\bar t,y)=0\}$ and D_x, D_t, $D_{(x,t)}$ denote the partial differentiation with respect to x, t and (x,t), respectively.

A vector ξ satisfying (ii) in the EMFCQ is called an EMF-vector at $\bar x\in M[H,G,\bar t\,]$. The original Mangasarian–Fromovitz constraint qualification was presented in [10] for the first time.

* Received by the editors October 28, 1992; accepted for publication (in revised form) October 26, 1993.

† Rheinisch-Westfälische Technische Hochschule Aachen, Lehrstuhl C für Mathematik, Templergraben 55, D-52062 Aachen, Germany.

‡ Humboldt-Universität zu Berlin, Fachbereich Mathematik, Institut für Angewandte Mathematik, PSF 1297, D-10099 Berlin, Germany.

In the present paper we continue our stability investigations of feasible sets. In previous papers [5], [9], [11], and [12], we have shown that the overall validity of the EMFCQ on the considered feasible set is closely related to its stability with respect to the describing function vector (H, G) belonging to the corresponding function space that is topologized by means of the strong (or Whitney-) C^1-topology (see [7], [8]). (In [5], respectively, [9], we considered nonparametric *compact* finite, i.e., Y is a finite set, respectively, semi-infinite, feasible sets; in [11], respectively, [12] we investigated *noncompact* one-parametric finite, respectively, nonparametric semi-infinite sets.)

These theoretical results yield consequences for establishing optimization algorithms and numerical applications, for example, within investigations of path-following algorithms in one-parametric optimization (see, e.g., [4]). In [11] we considered one-parametric feasible sets in finite optimization and proved a sufficient condition for the fact that $M[H, G, t_1]$ is homeomorphic with $M[H, G, t_2]$ for all $t_1, t_2 \in T$.

In the present investigations we extend the latter result for the given semi-infinite structure. To this aim we introduce the following specified global constraint qualification that contains additional parameter information.

DEFINITION. *The extended Mangasarian–Fromovitz constraint qualification at infinity* (EMFCQI) *is satisfied at* $M[H, G, T]$ *if there exists a real* $\gamma > 0$ *such that*

$$\inf \left\{ \begin{array}{l} \left\| \sum_{i \in I} \alpha_i Dh_i(\bar{x}, \bar{t}) + \sum_{j=1}^{r'} \beta_j D_{(x,t)} G(\bar{x}, \bar{t}, y^j) + \alpha_o \begin{pmatrix} 0_n \\ 1 \end{pmatrix} \right\| \\ \hline \alpha_i \in R,\ i = 0, \ldots, m,\ \beta_j \geq 0,\ y^j \in E_G(\bar{x}, \bar{t}),\ j = 1, \ldots, r',\ r' < \infty, \\ \sum_{i=0}^{m} |\alpha_i| + \sum_{j=1}^{r'} \beta_j = 1,\ (\bar{x}, \bar{t}) \in M[H, G, T] \end{array} \right\} > \gamma,$$

where 0_n *denotes the zero in* R^n *and* $\| \cdot \|$ *the Euclidean norm.*

Now, we can formulate our main result.

THEOREM. *Let the* EMFCQI *be satisfied at* $M[H, G, T]$. *Then, for any pair* $t_1, t_2 \in T$:

$$M[H, G, t_1] \simeq M[H, G, t_2]$$

($\simeq$ *means "is homeomorphic with"*).

The present paper is organized as follows. In §2 we present the lemmas used later for the proof of the Theorem as well as some properties related to the EMFCQI. In §3 we prove the Theorem and give some additional remarks.

2. Lemmas and first results. Throughout this paper we denote, as usual, the strong (or Whitney-) C^1-topology by C^1_s (for a detailed study on this subject, we refer to [7]). The following two facts are well known and were given in a previous paper [9].

LEMMA 1. EMFCQ *is invariant under local* C^1*-coordinate transformations.*

LEMMA 2. *Let* $U \subset R^n$ *be an open set. Then* $C^\infty(U, R)$ *is* C^1_s*-dense in* $C^1(U, R)$.

In Lemma 3 let $U \subset R^n$ be an open set and $F : U \to R^n$ a vector field. The lemma presents a known result from the theory of ordinary differential equations (see, e.g., [1]).

LEMMA 3. *If* F *is locally Lipschitzian, respectively,* $F \in C^k(U, R^n)$, *then there exists a continuous, respectively,* k*-times differentiable flow* ϕ *of* F, *which is defined on an open neighbourhood of* $U \times \{0\}$.

If, in particular, F *is locally Lipschitzian and bounded as well as* $U = R^n$, *then* ϕ *is defined on the whole space* $R^n \times R$, *i.e.,* F *is completely integrable.*

The subsequent lemma was proved in [9]. Without loss of generality we have chosen $m = 0$ (no equality constraints) since we consider a local property only.

LEMMA 4. *Let $\bar{t} \in T$ be fixed, $\bar{x} \in R^n$ and $\max_{y \in Y} G(\bar{x}, \bar{t}, y) = 0$. Furthermore, let $U \subset R^n$ be an open neighbourhood of $\bar{x}$, $F \in C^1(U, R^n)$, ϕ the flow of F, and $F(\bar{x})$ an* EMF*-vector at $\bar{x}$. Finally, suppose that the point $\hat{x}$ belongs to the same trajectory of ϕ as $\bar{x}$, i.e., $\hat{x} = \phi(\bar{x}, -\bar{v})$ for an integration time $\bar{v} \in R$.*

Then, there exists an open neighbourhood V of $\hat{x}$, a real number $\delta > 0$ and a unique Lipschitzian function $\tilde{T} : V \to [\bar{v} - \delta, \bar{v} + \delta]$ such that $\phi(x, v)$ is defined on $V \times [\bar{v} - \delta, \bar{v} + \delta]$ and, for $x \in V$, the function $\max_{y \in Y} G(\phi(x, v), \bar{t}, y)$ vanishes exactly at $v = \tilde{T}(x)$ and changes sign when passing $v = \tilde{T}(x)$.

For a given set $K \subset R^n$ we set

$$C(K) = \left\{ x \in R^n | x = \sum_{i=1}^{p} w_i s_i, \quad w_i \geq 0,\ s_i \in K,\ i = 1, \ldots, p, \quad p < \infty,\ \sum_{i=1}^{p} w_i = 1 \right\}$$

and

$$L(K) = \left\{ x \in R^n | x = \sum_{i=1}^{p} w_i s_i,\ w_i \in R,\ s_i \in K,\ i = 1, \ldots, p,\ p < \infty \right\}.$$

LEMMA 5 ([3]). *Let $K \subset R^n$ be a nonempty compact set. Then, the system*

$$z^T \xi < 0, \qquad z \in K$$

has a solution if and only if $0 \notin C(K)$.

We recall the Carathéodory theorem (see, e.g., [6]).

LEMMA 6. *Let $K \subset R^n$. Then, for each $x \in C(K)$ there exist $s_i \in K$ and $w_i > 0$, $i = 1, \ldots, p'$ such that*

$$p' \leq \dim L(K) + 1, \qquad \sum_{i=1}^{p'} w_i = 1 \quad \text{and} \quad x = \sum_{i=1}^{p'} w_i s_i,$$

where dim *denotes the dimension.*

Now, we present a list of three lemmas that are closely related to the above defined condition EMFCQI.

LEMMA 7. *Let $\bar{t} \in T$ and $\bar{x} \in M[H, G; \bar{t}]$. Then, the* EMFCQ *is satisfied at $\bar{x} \in M[H, G, \bar{t}]$ if and only if there exists a real $\gamma(\bar{x}, \bar{t}) > 0$ such that*

(2.1)

$$\inf \left\{ \frac{\left\| \sum_{i \in I} \alpha_i Dh_i(\bar{x}, \bar{t}) + \sum_{j=1}^{\bar{r}} \beta_j D_{(x,t)} G(\bar{x}, \bar{t}, y^j) + \alpha_o \begin{pmatrix} 0_n \\ 1 \end{pmatrix} \right\|}{\begin{array}{l} \alpha_i \in R,\ i = 0, \ldots, m,\ \beta_j \geq 0,\ y^j \in E_G(\bar{x}, \bar{t}),\ j = 1, \ldots, \bar{r}, \bar{r} < \infty, \\ \sum_{i=0}^{m} |\alpha_i| + \sum_{j=1}^{\bar{r}} \beta_j = 1, \end{array}} \right\} > \gamma(\bar{x}, \bar{t}).$$

Proof. Let the EMFCQ be satisfied at $\bar{x} \in M[H,G,\bar{t}]$. Then, for each nonzero $a \in R$, there exists a $\xi^a \in R^{n+1}$ solving the system

$$\begin{aligned} D_{(x,t)}G(\bar{x},\bar{t},y)\xi^a &< 0, \quad y \in E_G(\bar{x},\bar{t}), \\ (0_n^T, a)\xi^a &< 0. \end{aligned} \tag{2.2}$$

Now, suppose that (2.1) is not true. To avoid technical difficulties we restrict ourselves to the case of $m = 0$. Then, Lemma 6 yields the existence of sequences $\{\beta_1^\nu, \dots, \beta_{n+2}^\nu, \alpha_o^\nu\}_{\nu\in N}$, $\{y^{1,\nu}, \dots, y^{n+2,\nu}\}_{\nu\in N} \subset E_G(\bar{x},\bar{t})$ satisfying

$$\left\| \sum_{j=1}^{n+2} \beta_j^\nu D_{(x,t)}G(\bar{x},\bar{t},y^{j,\nu}) + \alpha_o^\nu \begin{pmatrix} 0_n \\ a \end{pmatrix} \right\| \to 0 \quad \text{for } \nu \to \infty,$$

where

$$\beta_j^\nu \geq 0,\ j = 1,\dots,n+2,\ \nu \in N, \alpha_o^\nu \geq 0,\ \nu \in N,\ \alpha_o^\nu + \sum_{j=1}^{n+2} \beta_j^\nu = 1,$$

$|a| = 1$, and $N = \{1,2,3,\dots\}$ denotes the set of natural numbers. Since Y is compact we can assume without loss of generality that all appearing sequences converge for $\nu \to \infty$. This fact, however, is in contradiction with Lemma 5 and (2.2).

The other direction follows immediately from Lemma 5. □

Two obvious consequences of the latter lemma are given in the following corollary.

COROLLARY 1. (i) *Let the* EMFCQI *be satisfied at* $M[H,G,T]$. *Then, the* EMFCQ *is satisfied at* $\bar{x} \in M[H,G,\bar{t}]$ *for all* $(\bar{x},\bar{t}) \in M[H,G,T]$.

(ii) *Let* $\bar{t} \in T$, $M[H,G,\bar{t}]$ *be compact and the* EMFCQ *be satisfied at each* $\bar{x} \in M[H,G,\bar{t}]$. *Then, there exists a* $\bar{\gamma} > 0$

$$\inf \left\{ \frac{\left\| \sum_{i\in I} \alpha_i Dh_i(\bar{x},\bar{t}) + \sum_{j=1}^{\bar{r}} \beta_j D_{(x,t)}G(\bar{x},\bar{t},y^j) + \alpha_o \begin{pmatrix} 0_n \\ 1 \end{pmatrix} \right\|}{\begin{array}{l} \alpha_i \in R,\ i = 0,\dots,m,\ \beta_j \geq 0, y^j \in E_G(\bar{x},\bar{t}),\ j = 1,\dots,\bar{r},\ \bar{r} < \infty, \\ \bar{x} \in M[H,G,\bar{t}], \sum_{i=0}^{m} |\alpha_i| + \sum_{j=1}^{\bar{r}} \beta_j = 1 \end{array}} \right\} > \bar{\gamma}.$$

LEMMA 8. *Let the* EMFCQI *be satisfied at* $M(H,G,T)$ *and* $a \in R$, $a \neq 0$ *arbitrarily chosen and fixed. Then, there exists a mapping*

$$\bar{F}^a : (x,t) \in M[H,G,T] \mapsto \bar{F}^a(x,t) \in R^{n+1}$$

satisfying

$$\begin{aligned} &Dh_i(x,t)\bar{F}^a(x,t) = 0, \qquad i \in I \\ &D_{(x,t)}G(x,t,y)\bar{F}^a(x,t) \leq 0,\ y \in E_G(x,t) \\ &\|\bar{F}^a(x,t)\| < \frac{|a|}{\gamma} \\ &\bar{F}_t^a(x,t) = a, \end{aligned}$$

for each $(x,t) \in M[H,G,T]$, *where* $\bar{F}^a(x,t)^T \in R^n \times R$, *and* $\bar{F}^a_t(x,t)$ *is the* $(n+1)$*th component of* $\bar{F}^a(x,t)$.

Proof. Suppose $M[H,G,T] \neq \emptyset$ and $(\bar{x},\bar{t}\,) \in M[H,G,T]$. Consider the optimization problem

$$\text{Minimize } \tfrac{1}{2}\|\xi\|^2 \text{ subject to } \left\{ \xi \in R^{n+1} \;\middle|\; \begin{array}{l} Dh_i(\bar{x},\bar{t}\,)\xi = 0,\ i \in I \\ D_{(x,t)}G(\bar{x},\bar{t},y)\xi \leq 0,\ y \in E_G(\bar{x},\bar{t}\,) \\ (0_n^T,1)\xi = a \end{array} \right\}.$$

The feasible set of this problem is convex and, due to the EMFCQI, nonempty. Then, the strict convexity of $\frac{1}{2}\|\xi\|^2$ and the generalized Kuhn–Tucker theorem for semi-infinite optimization problems (see, e.g., [6]) imply the existence of a unique solution $\bar{F}^a(\bar{x},\bar{t}\,)$ of the latter problem as well as finitely many multipliers $\tilde{\alpha}_i \in R$, $i = 0,\ldots,m$, $\tilde{\beta}_j \geq 0$, $j = 1,\ldots,\tilde{p}$, and $y^j \in \{y \in E_G(\bar{x},\bar{t}\,) | D_{(x,t)}G(\bar{x},\bar{t},y)\bar{F}^a(\bar{x},\bar{t}\,) = 0\}$, $j = 1,\ldots,\tilde{p}$ with $\bar{F}^a(\bar{x},\bar{t}\,) = \tilde{\alpha}_o\binom{0_n}{1} + \sum_{i\in I}\tilde{\alpha}_i Dh_i(\bar{x},\bar{t}\,) - \sum_{j=1}^{\tilde{p}}\tilde{\beta}_j D_{(x,t)}G(\bar{x},\bar{t},y^j)$. Multiplying the latter equation by $\bar{F}^a(\bar{x},\bar{t}\,)$ we obtain $\|\bar{F}^a(\bar{x},\bar{t}\,)\|^2 = \tilde{\alpha}_o \cdot a$ and, hence, $\tilde{\alpha}_o \neq 0$ as well as

$$\frac{|a|}{\|\bar{F}^a(\bar{x},\bar{t}\,)\|} = \left\| \frac{\tilde{\alpha}_o}{|\tilde{\alpha}_o|}\binom{0_n}{1} + \sum_{i\in I}\frac{\tilde{\alpha}_i}{|\tilde{\alpha}_o|}Dh_i(\bar{x},\bar{t}\,) - \sum_{j=1}^{\tilde{p}}\frac{\tilde{\beta}_j}{|\tilde{\alpha}_o|}D_{(x,t)}G(\bar{x},\bar{t},y^j) \right\| > \gamma.$$

Consequently, we have $\|\bar{F}^a(\bar{x},\bar{t}\,)\| < \frac{|a|}{\gamma}$. □

Note, that the mapping $\bar{F}^a$ in Lemma 8 need not be continuous. The proof of the next lemma is omitted since it uses only continuity arguments.

LEMMA 9. *Let the* EMFCQI *be satisfied at* $M[H,G,T]$. *Then, there exists a* C^1_s*-neighbourhood* ϑ' *of* (H,G) *in* $C^1(R^n \times R, R^m) \times C^1(R^n \times R \times R^r, R)$ *such that the* EMFCQI *is satisfied at* $M[H',G',T]$ *for all* $(H',G') \in \vartheta'$ *with the same* $\gamma > 0$.

LEMMA 10. *Let the* EMFCQ *be satisfied at* $\bar{x} \in M[H,G,\bar{t}\,]$ *for all* $(\bar{x},\bar{t}\,) \in M[H,G,T]$. *Then, there exists a* C^1_s*-neighbourhood* ϑ'' *of* (H,G) *in* $C^1(R^n \times R, R^m) \times C^1(R^n \times R \times R^r, R)$ *such that for all* $(H',G') \in \vartheta''$ *and each* $t \in T$:

$$M[H,G,t] \simeq M[H',G',t].$$

On the proof of the latter lemma we only note that the basic idea for the construction of the desired homeomorphism is a generalization of the corresponding homeomorphism which we used in the case with compact feasible sets and a finite set Y. This homeomorphism is in a precise way described in [5, Thm. B]. The generalization mentioned can be obtained by using Lemmas 3.5 and 3.6 in [9] as well as the construction given in detail in [11, Thm. 1].

3. Proof of the Theorem and remarks. *Proof of the Theorem.* The proof is given in four steps. A geometrical illustration is given later on in Remark 1.

Step 1. Lemmas 9 and 10 imply the existence of a C^1_s-neighbourhood ϑ of (H,G) in $C^1(R^n \times R, R^m) \times C^1(R^n \times R \times R^r, R)$ such that for each $(H',G') \in \vartheta$:

(i) the EMFCQI is satisfied at $M[H',G',T]$ with the same $\gamma > 0$ and

(ii) $M[H,G,t] \simeq M[H',G',t]$ for all $t \in T$.

Therefore, it suffices to show for a suitable $(H^*,G^*) \in \vartheta$ that $M[H^*,G^*,t_1] \simeq M[H^*,G^*,t_2]$ for arbitrarily chosen $t_1,t_2 \in T$. By Lemma 2, assume without loss of generality that $(H,G) \in C^3(R^n \times R, R^m) \times C^3(R^n \times R \times R^r, R)$, $\underline{T} = [0,1]$ and we show that

$$M[H,G,0] \simeq M[H,G,1].$$

Throughout the proof we denote $\underline{M} = M[H,G,T]$, $M(t) = M[H,G,t]$ for $t \in T$, $\underline{\partial M} = \{(x,t) \in M \mid \max_{y\in Y} G(x,t,y) = 0\}$ and $\underline{\partial M(t)} = \{x \in M(t) \mid (x,t) \in \partial M\}$ for $t \in T$.

Let $\bar{F}^a$ be defined as in Lemma 8. By Corollary 1, there exists an EMF-vector $\tilde{w}(\bar{x},\bar{t})$ at $\bar{x} \in M(\bar{t})$ whenever $(\bar{x},\bar{t}) \in M$. Adding the sufficiently small vector $(\tilde{w}(\bar{x},\bar{t}))$ to $\bar{F}^a(\bar{x},\bar{t})$, we obtain an $\tilde{F}^a(\bar{x},\bar{t})$ such that for all $(x,t) \in M$:

$$
\begin{aligned}
&Dh_i(x,t)\tilde{F}^a(x,t) = 0, \qquad i \in I,\\
&D_{(x,t)}G(x,t,y)\tilde{F}^a(x,t) < 0, \qquad y \in E_G(x,t),\\
&\tilde{F}^a_t(x,t) = a, \ (\text{where } \tilde{F}^a(x,t)^T = (\tilde{F}^a_x(x,t)^T, \tilde{F}^a_t(x,t)) \in R^n \times R),\\
&\|\tilde{F}^a(x,t)\| < \tfrac{|a|}{\gamma}.
\end{aligned} \tag{3.1}
$$

For $(\bar{x},\bar{t}) \in M$ and by using local coordinates we can construct a neighbourhood $U(\bar{x},\bar{t})$ of $(\bar{x},\bar{t})$ and an $\tilde{F}^a_{(\bar{x},\bar{t})} \in C^2(U(\bar{x},\bar{t}), R^{n+1})$ satisfying (3.1) for all $(x,t) \in U(\bar{x},\bar{t})$ if replacing there $\tilde{F}^a$ by $\tilde{F}^a_{(\bar{x},\bar{t})}$.

Now, on $R^{n+1}\backslash M$ select the constant vector field $\binom{0_n}{a}$ and on $U(\bar{x},\bar{t})$ even $\tilde{F}^a_{(\bar{x},\bar{t})}$. Then, by using a partition of unity subordinate to the covering $\{U(\bar{x},\bar{t}),\ (\bar{x},\bar{t}) \in M,\ R^{n+1}\backslash M\}$ of R^{n+1} we obtain a bounded vector field $\underline{F^a} \in C^2(R^{n+1}, R^{n+1})$ satisfying (3.1) on an open neighbourhood $\underline{U}$ of M if replacing $\tilde{F}^a$ by F^a in (3.1). Let $\underline{\psi^a((x,t),\tau)}$ and $\underline{\psi((x,t),\tau)}$ denote the flows of F^a and $(-F^{-1})$, respectively.

Step 2. Throughout this step let $\hat{T} \subset T$ be compact, $\partial M(\hat{T}) = \{(x,t) \in \partial M \mid t \in \hat{T}\}$, $V \subset U$ an open neighbourhood of $\partial M(\hat{T})$ and $w(x,t) \in C^1(V, R^{n+1})$ such that for all $(x,t) \in V$:

$$
\begin{aligned}
&Dh_i(x,t)w(x,t) = 0,\ i \in I,\\
&w(x,t)^T = (w_x(x,t)^T, 0) \in R^n \times R \quad \text{and}\\
&w_x(x,t) \text{ is an EMF-vector at } x \in \partial M(t) \text{ if } (x,t) \in \partial M.
\end{aligned}
$$

Furthermore, let $\phi((x,t),v)$ denote the flow of w. Obviously, there exists a continuous positive function $\beta(x,t)$ on $\partial M(\hat{T})$ such that $\phi((\bar{x},\bar{t}),\cdot)$ is defined on $(-\beta(\bar{x},\bar{t}), \beta(\bar{x},\bar{t}))$ for $(\bar{x},\bar{t}) \in \partial M(\hat{T})$.

Now, it is easily seen from Lemma 4 and a continuity argument that the following Contention 1 is true.

CONTENTION 1. *Let $W \subset \partial M(\hat{T})$ be open with respect to $\partial M(\hat{T})$. Then, the set $\{\phi((x,t),v) \mid (x,t) \in W,\ v \in (-\beta(\bar{x},\bar{t}), \beta(\bar{x},\bar{t}))\}$ is open with respect to $\{(x,t) \in R^n \times \hat{T} \mid h_i(x,t) = 0,\ i \in I\}$.*

In the remainder of this step let $\underline{A} \subset \partial M(\hat{T})$ be a nonempty compact set. From Lemma 4 we learn that there exists a $\bar{v} \in (0, \min\{\beta(x,t) \mid (x,t) \in A\}]$ and a unique real-valued Lipschitzian $\tilde{T}$ on $\tilde{N}_o = \{\phi((x,t),v) \mid (x,t) \in A,\ v \in (-\bar{v},\bar{v})\}$ with:

$$
\left.
\begin{aligned}
&h_i(\psi((\bar{x},\bar{t}), \tilde{T}(\bar{x},\bar{t}))) = 0,\ i \in I\\
&\max_{y\in Y} G(\psi((\bar{x},\bar{t}), \tilde{T}(\bar{x},\bar{t})), y) = 0
\end{aligned}
\right\} \quad \text{for } (\bar{x},\bar{t}) \in \tilde{N}_o,
$$

$$
\tilde{T}(x,t) = 0 \quad \text{for } (x,t) \in A,
$$

$$
\psi((\bar{x},\bar{t}),\tau) \in U \quad \text{for } (\bar{x},\bar{t}) \in \tilde{N}_o \quad \text{and} \quad \tau \in [0, \tilde{T}(\bar{x},\bar{t})].
$$

A careful, but stupid, calculation provides that Contention 2 is true.

CONTENTION 2. *There is a $\tilde{v} \in (0,\bar{v})$ such that for all $(x,t) \in A$:*

(i) *For all* $v \in [0, \tilde{v}]$ *the vector* $\partial_v \psi_x(\phi((x,t),v),\tau)|_{\tau=\tilde{T}(\phi((x,t),v))}$ *is an* EMF-*vector at*

$$\psi_x(\phi((x,t),v),\tilde{T}(\phi((x,t),v))) \in M(t+\tilde{T}(\phi((x,t),v))),$$

where

$$\psi = \begin{pmatrix} \psi_x \\ \psi_t \end{pmatrix} \quad \text{with } \psi_x \in R^n.$$

Obviously,

$$\partial_v \psi_t(\phi((x,t),v),\tau)|_{\tau=\tilde{T}(\phi((x,t),v))} = 0.$$

(ii) *For all* $v^1, v^2 \in [0, \tilde{v}]$, *we have*

$$\tilde{T}(\phi((x,t),v^1)) < \tilde{T}(\phi((x,t),v^2)) \text{ if } v^1 < v^2.$$

In the remainder of Step 2 assume that the following condition holds for all $(\bar{x}, \bar{t}) \in A$:

$$\begin{aligned} &\frac{D_{(x,t)}G(\bar{x},\bar{t},\bar{y})w(\bar{x},\bar{t})}{D_{(x,t)}G(\bar{x},\bar{t},\bar{y})F^{-1}(\bar{x},\bar{t})} > 1, \qquad \bar{y} \in E_G(\bar{x},\bar{t}) \quad \text{and} \\ &\|\lambda w(\bar{x},\bar{t}) - F^{-1}(\bar{x},\bar{t})\| < \frac{1}{\gamma} \quad \text{for all} \quad \gamma \in [0,1]. \end{aligned} \tag{3.2}$$

Obviously, (3.2) is fulfilled if we choose $w(x,t) = F^1(x,t) + F^{-1}(x,t)$. Then, a careful moment of reflection shows that (3.2) ensures that the following contention is true, too.

CONTENTION 3. *There is a* $\hat{v} \in (0, \tilde{v}]$ *such that for all* $(x,t) \in A$ *and all* $v \in [0, \hat{v}]$:

(i) *Condition* (3.2) *is fulfilled after replacing there* $(\bar{x}, \bar{t})$ *by*

$$\psi(\phi((x,t),v),\tilde{T}(\phi((x,t),v)))$$

and $w(\bar{x}, \bar{t})$ *by*

$$\partial_v \psi(\phi((x,t),v),\tau)|_{\tau=\tilde{T}(\phi((x,t),v))}.$$

(ii) $\|\psi(\phi((x,t),v),\tilde{T}(\phi((x,t),v))) - (x,t)\| \le \frac{\tilde{T}(\phi((x,t),v))}{\gamma}$.

The compactness of A implies the existence of a $\underline{\hat{t}} > 0$ satisfying $\tilde{T}(\phi((x,t),\hat{v})) > \hat{\tau}$ whenever $(x,t) \in A$, where $\underline{\hat{v}}$ is chosen as in Contention 3. Hence, there exists a continuous real-valued function

$$\underline{v} : ((\bar{x},\bar{t}),\tau) \in A \times [0,\hat{\tau}] \mapsto v(\bar{x},\bar{t},\tau) \in [0,\hat{v}) \tag{3.3}$$

satisfying $\tilde{T}(\phi((\bar{x},\bar{t}),v(\bar{x},\bar{t},\tau))) = \tau$ (see Fig. 1).

Let us summarize Step 2. For given compact sets $\hat{T} \subset T$ and $A \subset \partial M(\hat{T})$ and an EMF-vector field $w(x,t)$ defined on an open neighbourhood of $\partial M(\hat{T})$, we have constructed reals $\hat{v} > 0$, $\hat{\tau} > 0$ and the function v in (3.3) with the described properties if (3.2) is satisfied.

Step 3. For $k \in N$ define

$$\begin{aligned} U_k &= \{x \in R^n \mid \|x\| < k\}, \\ B_k &= \left\{(x,t) \in R^n \times T \mid \|x\| \le k + \frac{t}{\gamma}\right\}. \end{aligned}$$

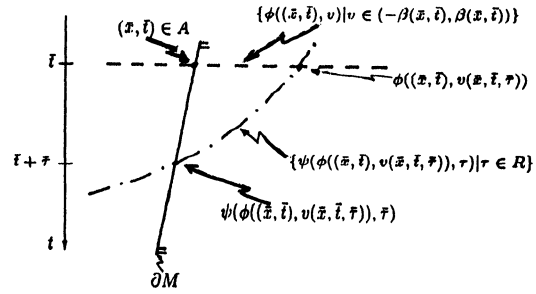


FIG. 1.

For $\hat{T} = T$, $A = B_k \cap \partial M$ and $w(x,t)$ satisfying (3.2) it follows that

$$\psi(\phi((\bar{x},\bar{t}),v(\bar{x},\bar{t},\tau)),\tau) \in A \tag{3.4}$$

for all $(\bar{x},\bar{t}) \in A$ and all $\tau \in [0,\hat{\tau}]$ if $\bar{t}+\tau \le 1$.

First, let $k = 2$. For $A = B_2 \cap \partial M$ and $w(x,t) = F^1(x,t) + F^{-1}(x,t)$ denote the corresponding terms $\hat{\tau} = \hat{\tau}^2$, $\hat{v} = \hat{v}^2$ and $v(x,t,\tau) = v^2(x,t,\tau)$. Then, we define

$$H^2 : (x,t) \in \operatorname{cl} U_2 \cap \partial M(0) \times T \mapsto H^2(x,t) \in \partial M$$

(where cl denotes the closure) via

$$H^2(x,t) = \begin{cases} (x,0) & \text{for } t = 0, \\ \psi(\phi((x,0),v^2(x,0,t)),t) & \text{for } t \in [0,\hat{\tau}^2], \\ \psi(\phi(H^2(x,\lambda\hat{\tau}^2),v^2(H^2(x, & \lambda\hat{\tau}^2),t-\lambda\hat{\tau}^2)),t-\lambda\hat{\tau}^2) \\ & \text{for } t \in [\lambda\hat{\tau}^2,\min\{1,(\lambda+1)\hat{\tau}^2\}], \\ & \lambda \in N, \lambda \ge 1 \text{ and } \lambda\hat{\tau}^2 < 1. \end{cases}$$

From (3.4) it follows that H^2 is well defined and, in particular, we have for $H^2(x,t)^T = (H_x^2(x,t)^T, H_t^2(x,t)) \in R^n \times T$, and $(x,t) \in \operatorname{cl} U_2 \cap \partial M(0) \times T$:

$$\begin{aligned} &\|H^2(x,t) - H^2(x,0)\| \le \frac{t}{\gamma}, \\ &H^2(x,t) \in \partial M,\ H_t^2(x,t) = t \text{ and, thus, } H_x^2(x,t) \in \partial M(t), \\ &H^2(x,t) \in B_2. \end{aligned} \tag{3.5}$$

It is easily seen from the construction that H^2 is *injective* and *continuous* on $\operatorname{cl} U_2 \cap \partial M(0) \times T$.

To extend the function H^2 we define the sets

$$V^1 = \{\psi(\phi((x,0),v),\tau)|v \in (-\hat{v}^2,\hat{v}^2),\tau \in [0,\hat{\tau}^2],\ x \in \partial M(0) \cap U_2\},$$
$$\tilde{V}^2 = \{\psi(\phi((x,0),v),\tau)|v \in [-v^2(x,0,\hat{\tau}^2),v^2(x,0,\hat{\tau}^2)],\tau \in [0,\hat{\tau}^2],x \in \partial M(0) \cap \operatorname{cl} U_1\},$$

and $V^2 = \{(x,t) \in R^n \times [0,\hat{\tau}^2]|h_i(x,t) = 0,\ i \in I\}\backslash\tilde{V}^2$. Then, $\{V^1,V^2\}$ is an open covering of $\{(x,t) \in R^n \times [0,\hat{\tau}^2]|h_i(x,t) = 0,\ i \in I\}$. Using a partition of unity subordinate to this latter covering and selecting on V^2 the vector field $F^1(x,t) + F^{-1}(x,t)$ and on V^1 the C^1-vector field (here, we need $F^a \in C^2$!), $\mathcal{F}$ defined as $\mathcal{F}(\psi(\phi((x,0),v),\tau)) =$

$\partial_v \psi(\phi((x,0),v),\tau)$ we obtain a C^1-vector field $w^{4.1}$ on $\{R^n \times [0,\hat{\tau}^2] | h_i(x,t) = 0,\ i \in I\}$ having the following properties:

$w^{4.1}(x,t)^T = (w_x^{4.1}(x,t),0) \in R^n \times R$;

$w_x^{4.1}(x,t)$ is an EMF-vector at $x \in \partial M(t)$;

$w^{4.1}$ satisfies (3.2) for $(x,t) \in \partial M$ replacing w by $w^{4.1}$ in (3.2); and for the flow $\phi^{4.1}$ of $w^{4.1}$ we have

$$\begin{aligned} &\phi^{4.1}((\bar{x},\bar{t}),w) = \psi(\phi((\tilde{x},0),\tilde{v}+w),\tilde{\tau}) \text{ if } (\bar{x},\bar{t}) \in \tilde{V}^2, \\ &(\bar{x},\bar{t}) = \psi(\phi((\tilde{x},0),\tilde{v}),\tilde{\tau}) \quad \text{and} \quad |\tilde{v}+w| \le v^2(\tilde{x},0,\hat{\tau}^2). \end{aligned} \tag{3.6}$$

By choosing $A = B_4 \cap \partial M([0,\hat{\tau}^2])$, $w(x,t) = w^{4.1}(x,t)$ and after determining the corresponding terms $\hat{\tau} = \hat{\tau}^{4.1}$, $\hat{v} = \hat{v}^{4.1}$, and $v(x,t,\tau) = v^{4.1}(x,t,\tau)$ we can define H^4 on $\operatorname{cl} U_4 \cap \partial M(0) \times [0,\hat{\tau}^2]$ in an analogous manner to the definition of H^2. The resulting function H^4 satisfies (3.5), after replacing there 2 by 4, and is injective and continuous on $\operatorname{cl} U_4 \cap \partial M(0) \times [0,\hat{\tau}^2]$. In particular, condition (3.6) implies

$$H^2|_{\operatorname{cl} U_1 \cap \partial M(0) \times [0,\hat{\tau}^2]} \equiv H^4|_{\operatorname{cl} U_1 \cap \partial M(0) \times [0,\hat{\tau}^2]}.$$

Extending H^4 in an analogous way to the construction on $\operatorname{cl} U_4 \cap \partial M(0) \times [0,\hat{\tau}^2]$ successively for $\lambda \in N$ with $\lambda\hat{\tau}^2 < 1$ on $\operatorname{cl} U_4 \cap \partial M(0) \times [\lambda\hat{\tau}^2, \min\{1,(\lambda+1)\hat{\tau}^2\}]$ we obtain the injective and continuous function

$$H^4 : (x,t) \in \operatorname{cl} U_4 \cap \partial M(0) \times T \mapsto H^4(x,t) \in \partial M$$

satisfying (3.5), after replacing there 2 by 4, and

$$H^2|_{\operatorname{cl} U_1 \cap \partial M(0) \times T} \equiv H^4|_{\operatorname{cl} U_1 \cap \partial M(0) \times T}.$$

Analogously, we can construct injective and continuous functions H^{2k}, $k = 3,4,5,\ldots$ on $\operatorname{cl} U_{2k} \cap \partial M(0) \times T$ satisfying (3.5), after replacing there 2 by $2k-$ and

$$H^{2k}|_{\operatorname{cl} U_{2k-1} \cap \partial M(0) \times T} \equiv H^{2k+2}|_{\operatorname{cl} U_{2k-1} \cap \partial M(0) \times T}.$$

Hence, we obtain an injective and continuous function

$$\underline{\mathcal{H} : \partial M(0) \times T \to \partial M}$$

via $\mathcal{H}(x,t) = H^{2k}(x,t)$, if $(x,t) \in \operatorname{cl} U_{2k-1} \cap \partial M(0) \times T$. Hence, for $(x,t) \in \partial M(0) \times T$:

$$\begin{aligned} &\|\mathcal{H}(x,t) - \mathcal{H}(x,0)\| \le \frac{t}{\gamma}, \\ &\mathcal{H}(x,t) \in \partial M,\ \mathcal{H}(x,t) = (\mathcal{H}_x(x,t),t), \mathcal{H}_x(x,0) = x. \end{aligned}$$

Step 4. The above construction of $\mathcal{H}$ implies that $\operatorname{Im}\mathcal{H} = \{\mathcal{H}(x,t) | (x,t) \in \partial M(0) \times T\}$ is open and closed with respect to ∂M. If $\mathcal{H}$ were not surjective then there would exist a connected component Z of ∂M with $Z \cap \operatorname{Im}\mathcal{H} = \emptyset$ and, thus, $Z \cap \partial M(\{0\}) = \emptyset$. An analogous procedure to the construction of $\mathcal{H}$ would create a path belonging to ∂M and connecting a point from Z with a point from $\partial M(\{0\})$. Therefore, $\mathcal{H}$ is surjective.

Now, we can adopt the argumentation from the finite case. On a neighbourhood of $\partial M(0)$ define the EMF-vector field $F_x^1(x,t) + F_x^{-1}(x,t) \in R^n$ with the corresponding flow $\tilde{\phi}(x,v)$ which is defined on $\{(x,v) | x \in \partial M(0),\ v \in (-3\alpha(x), 3\alpha(x))\}$ where α is a positive continuous function on $\partial M(0)$. For $\varrho \in (0,3]$ the set $N_\varrho = \{\tilde{\phi}(x,v) | x \in \partial M(0),\ v \in$

$(-\varrho\alpha(x), \varrho\alpha(x))\}$ is open with respect to $\{x \in R^n | h_i(x,0) = 0,\ i \in I\}$. By Lemma 4, there is a unique Lipschitzian $\tilde{S} : N_3 \to R$ with $\tilde{\phi}(x, \tilde{S}(x)) \in \partial M(0)$, $x \in N_3$ and, thus, $\hat{S}(x) = \tilde{\phi}(x, \tilde{S}(x))$ is continuous. Define the two *homeomorphisms*

$$Y_1 : M(0) \to M(0) \times \{0\} \cup \partial M \quad \text{and} \quad Y_2 : M(0) \times \{0\} \cup \partial M \to M(1)$$

via

$$Y_1(x) = \begin{cases} (x,0) & \text{for } x \in M(0)\backslash N_2, \\ (\tilde{\phi}(x, -2\alpha(\hat{S}(x)) - \tilde{S}(x)), 0) & \text{for } x \in (N_2\backslash N_1) \cap M(0), \\ \mathcal{H}\left(\hat{S}(x), 1 + \frac{\tilde{S}(x)}{\alpha(\hat{S}(x))}\right) & \text{for } x \in N_1 \cap M(0), \end{cases}$$

and $Y_2(x,t) = \psi_x^1((x,t), 1-t)$. Then, $\underline{Y_2 \circ Y_1} : M(0) \to M(1)$ is a *homeomorphism.* This completes the proof. □

Remark 1. In the following we illustrate the geometrical background of the given proof. The boundary strip $\{\tilde{\phi}(x,v) | x \in \partial M(0),\ v \in [0, 3\alpha(x))\}$ of $M(0)$ can be divided into the three strips $\{\tilde{\phi}(x,v) | x \in \partial M(0),\ v \in [0, \alpha(x))\}$, $\{\tilde{\phi}(x,v) | x \in \partial M(0),\ v \in [\alpha(x), 2\alpha(x))\}$, and $\{\tilde{\phi}(x,v) | x \in \partial M(0),\ v \in [2\alpha(x), 3\alpha(x))\}$, which are sketched in Fig. 2(b) by an interrupted line, a full line, and a dotted line along the trajectories, respectively. Since we have covered the set ∂M with disjoint paths $\{\mathcal{H}(\bar{x}, t) | t \in T\}$, $\bar{x} \in \partial M(0)$ (Fig. 2(a)), we can map the interrupted-line-strip onto ∂M by means of Y_1 by drawing $\{\tilde{\phi}(\bar{x}, v) | v \in [0, \alpha(\bar{x})]\}$ along $\{\mathcal{H}(\bar{x}, t) | t \in T\}$ and fastening the endpoints $(\bar{x}, 0)$ and $(\tilde{\phi}(\bar{x}, \alpha(\bar{x})), 0)$ at $\mathcal{H}(\bar{x}, 1)$ and $\mathcal{H}(\bar{x}, 0) = (\bar{x}, 0)$ (see Fig. 2(c)), respectively. Analogously, the full-line-strip is stretched onto $\{\tilde{\phi}(\bar{x}, v) | \bar{x} \in \partial M(0),\ v \in [0, 2\alpha(x)]\}$ (Fig. 2(c)). Furthermore, the difference set $M(0)\backslash\{\tilde{\phi}(\bar{x}, v) | \bar{x} \in \partial M(0),\ v \in [0, 2\alpha(x))\}$ remains constant under Y_1. Obviously, $Y_1 : M(0) \mapsto M(0) \times \{0\} \cup \partial M$ is a homeomorphism.

Now, each trajectory of the vector field F^1 that contains (exactly) one point $(\tilde{x}, \tilde{t})$ of $M(0) \times \{0\} \cup \partial M$ also contains (exactly) one point of $M(1) \times \{1\}$—namely $(Y_2(\tilde{x}, \tilde{t}), 1) = \psi^1((\tilde{x}, \tilde{t}),\ 1 - \tilde{t})$—and vice versa (Fig. 2(d)). So, $Y_2 : M(0) \times \{0\} \cup \partial M \to M(1)$ is a homeomorphism and the composition $Y_2 \circ Y_1 : M(0) \to M(1)$ maps $M(0)$ onto $M(1)$ homeomorphically.

Remark 2. The geometrical idea of the given proof is the same as in the case of finitely many inequality constraints (see [11, Thm. 2]). However, in the present paper we had to carry out quite a different construction of the paths $\{\mathcal{H}(\bar{x}, t), t \in T\}$, $\bar{x} \in \partial M(0)$ than in the finite case since the construction in the finite case uses this finiteness essentially.

Remark 3. Let the EMFCQI be satisfied at $M[H, G, T]$. Then, the Theorem and Lemma 10 imply the existence of a C_s^1-neighbourhood ϑ of (H, G) in $C^1(R^n \times R, R^m) \times C^1(R^n \times R \times R^r, R)$ such that

$$M[H, G, t_1] \simeq M[H', G', t_2]$$

for all $(H', G') \in \vartheta$ and all $t_1, t_2 \in T$.

Remark 4. Obviously, we can weaken the assumption of the Theorem that the EMFCQI is satisfied at $M[H, G, T]$, since the proof uses only the existence of the vector fields F^a, $|a| = 1$, which satisfy (3.1) for all (x, t) from an open neighbourhood of $M[H, G, T]$ if replacing $\tilde{F}^a$ by F^a in (3.1).

Remark 5. We consider the point-set mapping

$$M[H, G, \cdot] : t \in T \mapsto M[H, G, t].$$

The property (3.1) of the vector fields F^1 and F^{-1} yields the upper semicontinuity in the sense of Hausdorff (*usc*-H) and the lower semicontinuity in the sense of Berge (*lsc*-B) of $M[H, G, \cdot]$

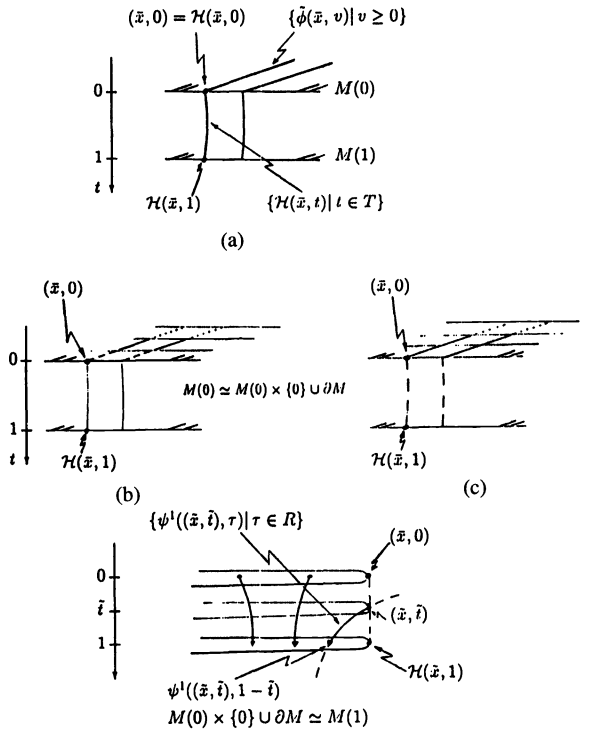


FIG. 2.

at each $t^\circ \in T$. Hence, $M[H,G,\cdot]$ is continuous at each $t^\circ \in T$ according to the notations in [2]. The proof of the final corollary is analogous to the proof of the corresponding corollary in the finite case ([11, Corollary 5]).

COROLLARY 2. *Let the* EMFCQI *be satisfied at* $M[H,G,T]$ *and* $(\bar{x},\bar{t}\,) \in M[H,G,T]$. *If* $|E_G(\bar{x},\bar{t}\,)|$ *is finite* ($|\cdot|$ *denotes the cardinality*), *then there exists an* EMF-*vector* $w(\bar{x},\bar{t}\,)$ *at* $\bar{x} \in M[H,G,\bar{t}\,]$ *satisfying* $\|w(\bar{x},\bar{t}\,)\| \leq |E_G(\bar{x},\bar{t}\,)|$ *and* $D_xG(\bar{x},\bar{t},y)w(\bar{x},\bar{t}\,) < -\gamma$, $y \in E_G(\bar{x},\bar{t}\,)$.

REFERENCES

[1] H. AMANN, *Gewöhnliche Differentialgleichungen*, Walter de Gruyter, Berlin, New York, 1983.

[2] B. BANK, J. GUDDAT, D. KLATTE, B. KUMMER, AND K. TAMMER, *Nonlinear Parametric Optimization*, Akademie-Verlag, Berlin, 1982.

[3] E. W. CHENEY, *Introduction to Approximation Theory*, McGraw-Hill, New York, 1966.

[4] J. GUDDAT, F. GUERRA VASQUEZ, AND H. TH. JONGEN, *Parametric Optimization: Singularities, Pathfollowing and Jumps*, John Wiley, New York, 1990.

[5] J. GUDDAT, H. TH. JONGEN, AND J. RÜCKMANN, *On stability and stationary points in nonlinear optimization*, J. Austral. Math. Soc., Ser. B 28 (1986), pp. 36–56.

[6] R. HETTICH AND P. ZENCKE, *Numerische Methoden der Approximation und semi-infiniten Optimierung*, Teubner Studienbücher, Stuttgart, 1982.

[7] M. W. HIRSCH, *Differential Topology*, Springer-Verlag, New York, Berlin, 1976.

[8] H. TH. JONGEN, P. JONKER, AND F. TWILT, *Nonlinear Optimization in* $\mathbb{R}^n$, II. *Transversality, Flows, Parametric Aspects*, Peter Lang Verlag, Frankfurt a.M., Bern, New York, 1986.

[9] H. TH. JONGEN, F. TWILT, AND G.-W. WEBER, *Semi-infinite optimization: Structure and stability of the feasible set*, J. Optim. Theory Appl., 72 (1992), pp. 529–552.

[10] O. L. MANGASARIAN AND S. FROMOVITZ, *The Fritz John necessary optimality conditions in the presence of equality and inequality constraints*, J. Math. Anal. Appl., 17 (1967), pp. 37–47.

[11] J.-J. RÜCKMANN, *Stability of noncompact feasible sets in nonlinear optimization*, in Parametric Optimization and Related Topics III, J. Guddat, H. Th. Jongen, B. Kummer, and F. Nožička, eds., Peter Lang Verlag, Frankfurt a.M., Bern, New York, 1993, pp. 467–502.

[12] J.-J. RÜCKMANN AND G.-W. WEBER, *Semi-infinite optimization: excisional stability of the feasible set*, J. Optim. Theory Appl., submitted.

SIAM J. OPTIMIZATION
Vol. 4, No. 3, pp. 649–658, August 1994
© 1994 Society for Industrial and Applied Mathematics
013

QUASIMONOTONICITY OF SEPARABLE OPERATORS AND MONOTONICITY INDICES*

J.-P. CROUZEIX† AND A. HASSOUNI†

Abstract. This research concerns the conditions that ensure the quasimonotonicity of the separable operator

$$F(x_1, x_2, \ldots, x_p) = (F_1(x_1), F_2(x_2), \ldots, F_p(x_p)),$$

where for $i = 1, 2, \ldots, p$, C_i is an open convex subset of $\mathbb{R}^{n_i}$ and $F_i : C_i \to \mathbb{R}^{n_i}$ is a continuous nonnull operator. It is shown, in particular, that all F_i, except perhaps one, are monotone. The conditions are given in terms of the monotonicity indices of the operators F_i, a concept introduced in this paper.

Key words. monotonicity, generalized monotonicity, separability, index of monotonicity and convexity

AMS subject classifications. 90C26, 90C33

1. Introduction. For $i = 1, 2, \ldots, p$, let C_i be an open convex set of $\mathbb{R}^{n_i}$ and $F_i : C_i \to \mathbb{R}^{n_i}$. We consider the operator F defined on the product set $C = C_1 \times C_2 \times \ldots \times C_p$ by

$$F(x_1, x_2, \ldots, x_p) = (F_1(x_1), F_2(x_2), \ldots, F_p(x_p)).$$

We are concerned with the quasimonotonicity of F on C.

If for $i = 1, 2, \ldots, p$, F_i is the gradient of a function f_i, then the quasimonotonicity of F is equivalent to the quasiconvexity of the function f:

$$f(x_1, x_2, \ldots, x_p) = f_1(x_1) + f_2(x_2) + \cdots + f_p(x_p).$$

Then it is known that all f_i, except perhaps one, are convex (Yaari [9], Debreu and Koopmans [3], Crouzeix and Lindberg [2]). It follows then that all operators F_i, except perhaps one, are monotone. We propose to extend this result to a more general case where F_i are not necessarily gradients of functions. To simplify the analysis, we assume that F_i are continuous.

Debreu and Koopmans and Crouzeix and Lindberg used the concept of convexity indices in their approaches. Similarly, we introduce here monotonicity indices. When F is the gradient of a function f, the monotonicity index of F equals the convexity index of f (this occurs particularly when F is operator defined on an interval of $\mathbb{R}$). The difficulties appear when the operators are not gradients of functions. Nevertheless, via these indices, we can transpose the results on the quasiconvexity of separable functions to the quasimonotonicity of operators.

The following questions remain open and will be the object of a second paper.

1. What happens when F_i are not continuous?

2. What happens when the results are transposed in the case where F_i are multivalued operators?

Now a few words on the notation used in this paper. $F : C \to \mathbb{R}^n$ is said to be monotone on C if for all $x_1, x_2 \in C$,

$$\langle F(x_2) - F(x_1), x_2 - x_1 \rangle \geq 0,$$

pseudomonotone, on C if for all $x_1, x_2 \in C$

$$\langle F(x_1), x_2 - x_1 \rangle > 0 \Rightarrow \langle F(x_2), x_2 - x_1 \rangle > 0,$$

* Received by the editors December 22, 1992; accepted for publication November 24, 1993.

† Applied Mathematics, Université Blaise Pascal and U.A. Centre National de la Recherche Scientifique (Paris) 1501, 63177 Aubiere Cedex, France.

quasimonotone on C if for all $x_1, x_2 \in C$

$$\langle F(x_1), x_2 - x_1\rangle > 0 \Rightarrow \langle F(x_2), x_2 - x_1\rangle \geq 0.$$

For more details on pseudomonotone, quasimonotone operators and their applications, see, for instance, Hassouni [4] and [5], Karamardian [6], and Karamardian and Schaible [7]. For a first order characterization see Karamardian, Schaible, and Crouzeix [8].

2. The convexity index of a function. Convexity indices were first introduced by Debreu and Koopmans [3] and next revisited by Crouzeix and Lindberg [2] for the study of additively quasiconvex functions. We shall use the Crouzeix–Lindberg definition.

Let C be an open convex subset of $\mathbb{R}^n$ and $f : C \to \mathbb{R}$. For all $\lambda \in \mathbb{R}$, set $f_\lambda = \exp(-\lambda f)$. Then $c(f)$, the convexity index of f, is defined as follows:

If there exists $\lambda < 0$ such that f_λ is not convex, then $c(f) = \mathrm{Sup}[\lambda : \lambda < 0,\ f_\lambda \text{ is convex}]$; if not, $c(f) = \mathrm{Sup}[\lambda : \lambda \geq 0,\ f_\lambda \text{ is concave}]$.

The convexity index is an indicator of convexity in the sense that f is convex if and only if $c(f) \geq 0$. Furthermore, $c(f) > -\infty$ implies that f is pseudoconvex (actually f is r-convex following Avriel's terminology [1]); the converse is false.

A function f is convex (quasiconvex, pseudoconvex) on the convex C if and only if the restrictions of f to the lines that intersect C are convex (quasiconvex, pseudoconvex). This property presents a valuable interpretation in terms of the convexity index. Namely, for $a \in C$, $d \in \mathbb{R}^n$, let $f_{a,d}(t) = f(a + td)$, the function of one real variable that is defined on the interval $I_{a,d}(t) = \{t : a + td \in C\}$. Then

$$c(f) = \operatorname*{Inf}_{a,d} c(f_{a,d}). \tag{2.1}$$

Note also that for all $k > 0$

$$c(kf) = \frac{1}{k} c(f). \tag{2.2}$$

The importance of convexity indices appears in the following result.

THEOREM 2.1 (Crouzeix–Lindberg [2, Thms. 8 and 9]). *Let $C_1, C_2, \ldots, C_p$ be open convex subsets of $\mathbb{R}^{n_1}, \mathbb{R}^{n_2}, \ldots, \mathbb{R}^{n_p}$ $(p \geq 2)$. For $i = 1, 2, \ldots, p$, let $f_i : C_i \to \mathbb{R}$; f_i nonconstant on C_i. Now, for all $x = (x_1, x_2, \ldots, x_p) \in C = C_1 \times C_2 \times \cdots \times C_p$, let*

$$s(x) = f_1(x_1) + f_2(x_2) + \cdots + f_p(x_p).$$

Then s is quasiconvex on C if and only if one of the two following conditions holds:

(i) *all f_i are convex (i.e., $c(f_i) \geq 0$ for all i),*

(ii) *all f_i except one are convex and*

$$\frac{1}{c(f_1)} + \frac{1}{c(f_2)} + \cdots + \frac{1}{c(f_p)} \leq 0.$$

In both cases, the convexity index of s is given by the formula

$$\frac{1}{c(s)} = \frac{1}{c(f_1)} + \frac{1}{c(f_2)} + \cdots + \frac{1}{c(f_p)} \tag{2.3}$$

with the convention $\frac{1}{0} = +\infty$.

We shall use also a two-factor version of this theorem.

THEOREM 2.2 (Crouzeix–Lindberg, [2, Thm. 5]). *On the same assumptions as those in Theorem 2.1 but with $p = 2$, s is quasiconvex if and only if $c(f_1) + c(f_2) \geq 0$.*

It follows that if s is quasiconvex, then at least one of the functions is convex, and the other one is pseudoconvex.

For all these results and complements, the reader is referred to Crouzeix–Lindberg [2].

3. The monotonicity index of an operator. Let C be an open convex subset of $\mathbb{R}^n$ and $F : C \to \mathbb{R}^n$. As in the above section, define for all $a \in C$, $d \in \mathbb{R}^n$ the interval $I(a,d) = \{t : a + td \in C\}$. Next define on this interval the function of one real variable

$$F_{a,d}(t) = \langle F(a+td), d\rangle.$$

Then F is monotone (quasimonotone, pseudomonotone) if and only if all functions $F_{a,d}$ are monotone (quasimonotone, pseudomonotone). This consideration added to relation (2.1) motivates the quest of a monotonicity index for functions of one real variable, this index being extended to operators by a relation similar to (2.1). For reasons of simplicity, we consider only continuous operators in this first approach of the problem.

Let I be an open interval of $\mathbb{R}$ and $\phi : I \to \mathbb{R}$, ϕ continuous on I. Fix some $t_0 \in I$ and consider for all $t \in I$

$$\varphi(t) = \int_{t_0}^{t} \phi(s)ds. \tag{3.1}$$

Then we define $m(\phi)$, the monotonicity index of ϕ by

$$m(\phi) = c(\varphi). \tag{3.2}$$

Referring to the definition of $c(\varphi)$, it is clear that the definition does not depend on t_0 the lower bound of the integral. Furthermore, ϕ is the derivative of φ and consequently

$$\varphi'_\lambda(t) = -\lambda\phi(t)\exp(-\lambda\varphi(t)) = -\lambda\phi(t)\varphi_\lambda(t).$$

Since φ_λ is convex (concave) if and only if φ'_λ is nondecreasing (nonincreasing), it follows from the definition of $c(\varphi)$ that

$$m(\phi) = \operatorname{Sup}[\lambda \in \mathbb{R} : t \to \phi_\lambda(t) \text{ is nondecreasing on } I], \tag{3.3}$$

where $\phi_\lambda(t) = \phi(t)\exp(-\lambda\int_{t_0}^{t}\phi(s)ds)$.

It is worth noting the coherence of this definition. Indeed, it follows from properties of functions φ_λ (see Proposition 1 of Crouzeix–Lindberg) that ϕ_λ is nondecreasing, if and only if for all $\mu < \lambda$ ϕ_μ is nondecreasing. In particular, $\phi_{m(\phi)}$ is nondecreasing.

On the other hand, $m(\phi) \geq 0$ if and only if φ is convex. Since $\phi = \varphi'$, $m(\phi)$ is nonnegative if and only if ϕ is monotone. Similarly, $m(\phi) > -\infty$ implies that ϕ is pseudomonotone.

Assume now that C is an open convex set of $\mathbb{R}^n$ and $F : C \to \mathbb{R}^n$ is continuous on C. Then similarly to (2.1) we define

$$m(F) = \operatorname*{Inf}_{a,d} m(F_{a,d}). \tag{3.4}$$

Note that according to the remark on the lower bound of the integral in expression (3.1), we have for all $a \in C$, $d \in \mathbb{R}^n$ and $t \in \mathbb{R}$ such that $a + td \in C$

$$m(F_{a+td,d}) = m(F_{a,d}). \tag{3.5}$$

On the other hand, for all $k \in \mathbb{R}$

$$F_{a,kd}(t) = kF_{a,d}(kt).$$

It follows that for all $k \neq 0$

$$f_{a,kd}(t) = \int_0^t F_{a,kd}(s)ds = f_{a,d}\left(\frac{t}{k}\right). \tag{3.6}$$

Hence for all $a \in C$, $d \in \mathbb{R}^n$ and $k \in \mathbb{R}$, $k \neq 0$

$$m(F_{a,kd}) = m(F_{a,d}). \tag{3.7}$$

From relation (3.4) it follows that F is monotone if and only if $m(F) \geq 0$. Furthermore, $m(F) > -\infty$ implies that F is pseudomonotone.

If F is null, then $F_{a,d}$ is null and $f_{a,d}$ is constant for all (a, d). Hence

$$+\infty = c(f_{a,d}) = m(F_{a,d}) \quad \text{and} \quad m(F) = +\infty.$$

Conversely if $m(F) = +\infty$, then for all (a, d), $m(F_{a,d}) = c(f_{a,d}) = +\infty$ and $f_{a,d}$ is constant. It follows that first $F_{a,d}$ is null and next F is null.

On the other hand, it follows from relations (2.2), (3.2), and (3.4) that for all $k > 0$,

$$m(kF) = \frac{1}{k} m(F). \tag{3.8}$$

In the particular case where F is the gradient of a function, i.e., if there exists $f : C \to \mathbb{R}$ such that $F = \nabla f$, then we have $c(f) = m(F)$. Indeed, the function $f_{a,d}$ that appears in relation (3.6) is then exactly the restriction of the function f on the line $\{a + td, t \in \mathbb{R}\}$. This property shows the links between the convexity and the monotonicity indices. In the following sections we show that the nice properties of convexity indices of functions can be transposed to monotonicity indices of operators even if these are not gradients of functions.

4. The case of two operators. In this section, we assume that C is an open convex set of $\mathbb{R}^n$, D is an open convex set of $\mathbb{R}^p$, $F : C \to \mathbb{R}^n$, $G : D \to \mathbb{R}^p$, F and G are continuous and nonnull, C and D are not empty. We consider $H : C \times D \to \mathbb{R}^{n+p}$ be defined by

$$H(x_1, x_2) = (F(x_1), G(x_2)) \quad (x_1, x_2) \in C \times D.$$

THEOREM 4.1. *H is quasimonotone on $C \times D$ if and only if $m(F) + m(G) \geq 0$.*

Proof. We use the fact that the quasimonotonicity of H can be termed via the quasimonotonicity of functions of one real variable. Namely, let

$$x = (x_1, x_2) \in C \times D \quad \text{and} \quad d = (d_1, d_2) \in \mathbb{R}^{n+p}.$$

We are concerned with the function

$$\begin{aligned} H_{x,d}(t) &= \langle H(x_1 + td_1, x_2 + td_2), (d_1, d_2) \rangle \\ &= \langle F(x_1 + td_1), d_1 \rangle + \langle G(x_2 + td_2), d_2 \rangle \\ &= F_{x_1,d_1}(t) + G_{x_2,d_2}(t). \end{aligned}$$

It follows that

$$\begin{aligned} h_{x,d}(t) &= \int_0^t H_{x,d}(s)ds \\ &= \int_0^t F_{x_1,d_1}(s)ds + \int_0^t G_{x_2,d_2}(s)ds \\ h_{x,d}(t) &= f_{x_1,d_1}(t) + g_{x_2,d_2}(t). \end{aligned}$$

Assume that $m(F) + m(G) \geq 0$. Then since

$$m(F) + m(G) \leq m(F_{x_1,d_1}) + m(G_{x_2,d_2}) = c(f_{x_1,d_1}) + c(g_{x_2,d_2}),$$

the function of two real variables

$$\theta_{x,d}(t,s) = f_{x_1,d_1}(t) + g_{x_2,d_2}(s)$$

is quasiconvex. This is due to Theorem 2.2 when both functions f_{x_1,d_1} and g_{x_2,d_2} are not constant. If one of them, say, for instance, g_{x_2,d_2}, is constant, then $c(\theta_{x,d}) = c(f_{x_1,d_1}) > -\infty$ and here again $\theta_{x,d}$ is quasiconvex.

Hence we deduce that the function of one real variable $h_{x,d}$ is quasiconvex, and next that $H_{x,d}$ is quasimonotone. This is true for all $(x,d) \in (C \times D) \times \mathbb{R}^{n+p}$ and therefore H is quasimonotone.

Conversely, assume $c(F) + c(G) < 0$, and show that H is not quasimonotone. There exist $(x_1,d_1) \in C \times \mathbb{R}^n$ and $(x_2,d_2) \in D \times \mathbb{R}^p$ such that

$$c(f_{x_1,d_1}) + c(g_{x_2,d_2}) = m(F_{x_1,d_1}) + m(G_{x_2,d_2}) < 0.$$

Set $\alpha = f_{x_1,d_1}$, $\beta = g_{x_2,d_2}$ and consider the function of two real variables

$$\theta(u_1,u_2) = \alpha(u_1) + \beta(u_2);$$

then θ is not quasiconvex. It follows that (u_1,v_1) and $(u_2,v_2) \in \mathbb{R}^2$ exist such that the function of one real variable

$$\mu(t) = \theta(u_1 + tv_1, u_2 + tv_2)$$

is not quasiconvex.

Then take

$$\bar{x} = (x_1 + u_1 d_1, x_2 + u_2 d_2),$$
$$\bar{d} = (v_1 d_1, v_2 d_2),$$

and consider the functions $H_{\bar{x},\bar{d}}$ and $h_{\bar{x},\bar{d}}$.

It is clear that $h_{\bar{x},\bar{d}} = \mu$. Hence $H_{\bar{x},\bar{d}}$ is not quasimonotone and therefore H is also not quasimonotone. □

COROLLARY 4.2. *It holds that*

$$m(H) \leq \min(m(F), m(G)).$$

Proof. Consider the equality

$$H_{x,d}(t) = F_{x_1,d_1}(t) + G_{x_2,d_2}(t),$$

which appears in the beginning of the proof of the above theorem. Take $d_2 = 0$. Then

$$H_{x,d}(t) = F_{x_1,d_1}(t).$$

Hence, we deduce $m(H) \leq m(F)$. □

COROLLARY 4.3. *Let C be an open convex set of $\mathbb{R}^n$, $F : C \to \mathbb{R}^n$, F continuous on C.*

(i) *F is monotone on C if and only if the operator $H(x,y) = (F(x), F(y))$ is quasimonotone on $C \times C$.*

(ii) *Let I be any nonempty open interval of $\mathbb{R}$. Then F is monotone on C if and only if the operator $H(x,t) = (F(x), 1)$ is quasimonotone on $C \times I$.*

Proof. (i) is an immediate consequence of Theorem 4.1.

(ii) Take $G(t) = 1$. Then $m(G) = 0$. Apply the theorem. □

An immediate consequence of Theorem 4.1 is that at least one of the two operators is monotone when H is quasimonotone. The second operator is at least pseudomonotone. This can be generalized by a trivial argument in the case where more than two operators are involved. All of the operators are monotone except perhaps one. To elaborate we are led to study the monotonicity index of monotone separable operators.

5. The monotonicity index of a monotone separable operator. We assume that F, G, and H are defined as in §4.

THEOREM 5.1. *If F and G are monotone then*

$$\frac{1}{m(H)} = \frac{1}{m(F)} + \frac{1}{m(G)}$$

with the convention $\frac{1}{0} = +\infty$.

Proof. By assumption, H is monotone. Hence, $m(H) \geq 0$. If $m(F) = 0$ ($m(G) = 0$) then $m(H) = 0$ and the equality holds. On the other hand, F and G are nonnull so that we assume henceforth that

$$0 < m(F) < +\infty \quad \text{and} \quad 0 < m(G) < +\infty.$$

In the sequel, we use the notation introduced in the proof of Theorem 4.1 and we set

$$\frac{1}{\lambda} = \frac{1}{m(F)} + \frac{1}{m(G)}.$$

Now, let any $(x, d) \in (C \times D) \times \mathbb{R}^{n+p}$ and consider

$$\theta(t) = \exp(-\lambda h_{x,d}(t)) = r^{\frac{\lambda}{m(F)}}(t) s^{\frac{\lambda}{m(G)}}(t),$$

where

$$r(t) = \exp(-m(F) f_{x_1,d_1}(t)) \quad \text{and} \quad s(t) = \exp(-m(G) g_{x_2,d_2}(t)).$$

Since $m(F) \leq c(f_{x_1,d_1})$ and $m(G) \leq c(g_{x_2,d_2})$, we deduce that r and s are concave. Now θ is concave because

$$\frac{\lambda}{m(F)} + \frac{\lambda}{m(G)} = 1.$$

It follows that $c(h_{x,d}) = m(H_{x,d}) \geq \lambda$. Hence $m(H) \geq \lambda$.

Consider now $\mu > \lambda$. Then there exist (x_1, d_1) and (x_2, d_2) such that

$$m(F) \leq m(F_{x_1,d_1}) = c(f_{x_1,d_1}) < \frac{\mu}{\lambda} m(F),$$
$$m(G) \leq m(G_{x_2,d_2}) = c(g_{x_2,d_2}) < \frac{\mu}{\lambda} m(G).$$

Set $\theta(u, v) = f_{x_1,d_1}(u) + g_{x_2,d_2}(v)$. Relation (2.3) implies

$$\frac{1}{c(\theta)} > \frac{\lambda}{\mu m(F)} + \frac{\lambda}{\mu m(G)} = \frac{1}{\mu}.$$

This inequality implies that $\exp(-\mu\theta)$ is not concave. Hence there are (u_1, v_1) and (u_2, v_2) in $\mathbb{R}^2$ such that the function of one real variable

$$\beta(t) = \exp(-\mu\theta(u_1 + tv_1, u_2 + tv_2))$$

is not concave.

As in the proof of Theorem 4.1, set now

$$\bar{x} = (x_1 + u_1 d_1, x_2 + u_2 d_2),$$
$$\bar{d} = (v_1 d_1, v_2 d_2)$$

and consider again functions $H_{\bar{x},\bar{d}}$ and $h_{\bar{x},\bar{d}}$. Then

$$h_{\bar{x},\bar{d}}(t) = \theta(u_1 + tv_1, u_2 + tv_2).$$

The nonconcavity of β implies that

$$\mu > c(h_{\bar{x},\bar{d}}) = m(H_{\bar{x},\bar{d}}) \geq m(H).$$

Because this inequality holds for all $\mu > \lambda$, we have $\lambda \geq m(H)$ and the proof is complete. □

Assume now that more than two operators are involved. Namely, for $i = 1, 2, \ldots, p$, let C_i be an open convex set of $\mathbb{R}^{n_i}$, $F_i : C_i \to \mathbb{R}^{n_i}$ be a nonnull continuous operator. We consider F the operator defined on $C = C_1 \times C_2 \times \cdots \times C_p$ by

$$F(x) = (F_1(x_1), F_2(x_2), \ldots, F_p(x_p)).$$

Then, by an immediate induction, Theorem 5.1 is generalized in the following theorem.

THEOREM 5.2. *Assume that for $i = 1, 2, \ldots, p$, F_i is monotone. Then F is monotone and*

$$\frac{1}{m(F)} = \frac{1}{m(F_1)} + \frac{1}{m(F_2)} + \cdots + \frac{1}{m(F_p)}.$$

6. The general case. We can now establish the transposed version of Theorem 2.1 in terms of operators.

THEOREM 6.1. *Let $C_1, C_2, \ldots, C_p$ be open convex subsets of $\mathbb{R}^{n_1}, \mathbb{R}^{n_2}, \ldots, \mathbb{R}^{n_p}$ $(p \geq 2)$. For $i = 1, 2, \ldots, p$ let $F_i : C_i \to \mathbb{R}^{n_i}$, F_i continuous and nonnull on C_i. Now, let $F : C_1 \times C_2 \times \cdots \times C_p \to \mathbb{R}^{n_1+n_2+\cdots+n_p}$ be defined by*

$$F(x_1, x_2, \ldots, x_p) = (F_1(x_1), F_2(x_2), \ldots, F_p(x_p)).$$

Then F is quasimonotone on C if and only if one of the two following conditions holds:

(i) *All F_i are monotone.*

(ii) *All F_i except one are monotone and*

$$\frac{1}{m(F_1)} + \frac{1}{m(F_2)} + \cdots + \frac{1}{m(F_p)} \leq 0.$$

In both cases, the monotonicity index of F is given by the formula

$$\frac{1}{m(F)} = \frac{1}{m(F_1)} + \frac{1}{m(F_2)} + \cdots + \frac{1}{m(F_p)}. \tag{6.1}$$

Proof. It is clear that if F is quasimonotone, then all F_i except perhaps one are monotone. The first part of the proof is then a direct consequence of Theorems 4.1 and 5.2.

It remains to prove formula (6.1). We proceed similarly as for Theorem 9 in Crouzeix and Lindberg [2]. If all F_i are monotone, then (6.1) follows from Theorem 5.2. Assume that F_p is not monotone and consider for $k > 0$ the operator G_k defined on $I = (0, +\infty)$ by

$$G_k(t) = -\frac{k}{t}.$$

Then G_k is monotone and $m(G_k) = \frac{1}{k}$.

Define now on $C_1 \times C_2 \times \cdots \times C_p \times I$ the operator

$$T(x_1, x_2, \ldots, x_p, t) = (F_1(x_1), F_2(x_2), \ldots, F_p(x_p), G_k(t)).$$

Then by the first part of this theorem, T is quasimonotone if and only if

$$\frac{1}{m(F)} + \frac{1}{m(G_k)} \leq 0$$

or, again, using the same result, if and only if

$$\frac{1}{m(F_1)} + \frac{1}{m(F_2)} + \cdots + \frac{1}{m(F_p)} + \frac{1}{m(G_k)} \leq 0.$$

Formula (6.1) straightforwardly follows. □

7. Explicit formulae for the monotonicity index of a pseudomonotone operator. Throughout all this section, C is an open convex set of $\mathbb{R}^n$, $F : C \to \mathbb{R}^n$ is a continuous and pseudomonotone operator ($m(F) = -\infty$ if F is not pseudomonotone). A first formula is obtained by expliciting formulae (3.4) and (3.3).

THEOREM 7.1. *Local index of monotonicity.*

$$m(F) = \operatorname*{Inf}_{a \in C} m(F, a)$$

where

$$m(F, a) = \operatorname*{Inf}_{d} (\lambda(a, d) : \langle F(a), d\rangle > 0)$$

and

$$\lambda(a, d) = \frac{1}{\langle F(a), d\rangle} \liminf_{t \downarrow 0} \frac{1}{t} \ln \left(\frac{\langle F(a + td), d\rangle}{\langle F(a), d\rangle} \right).$$

Proof. (i) Let $a \in C$ and d be such that $\langle F(a), d\rangle > 0$, and we set $\phi(t) = \langle F(a+td), d\rangle$. Since F is pseudomonotone, $\phi(t)$ is positive for all $t > 0$.

It follows from relations (3.3) and (3.4) that $\phi_{m(F)}$ is nondecreasing. In particular for all $t > 0$,

$$0 < \phi(0) \leq \phi(t) \exp\left(-m(F) \int_0^t \phi(s) ds \right)$$

and

$$m(F) \int_0^t \phi(s) ds \leq \ln \frac{\phi(t)}{\phi(0)}.$$

Pass to the limit when $t \downarrow 0$. Then

$$m(F) \leq \frac{1}{\phi(0)} \liminf_{t \downarrow 0} \frac{1}{t} \ln \frac{\phi(t)}{\phi(0)},$$

i.e., $m(F) \leq \lambda(a, d)$.

(ii) Conversely, let $\lambda > m(F)$. Then there exist $b \in C$ and $d \in \mathbb{R}^n$ such that $\lambda > m(F_{b,d})$. Set ϕ and χ be defined by

$$\phi(t) = \langle F(b + td), d\rangle,$$
$$\chi(t) = \phi(t) \exp\left(-\lambda \int_0^t \phi(s) ds \right).$$

Then χ is not nondecreasing. Namely, there exists t_1, t_2 in $\mathbb{R}$ such that

$$t_1 < t_2 \quad \text{and} \quad \chi(t_1) > \chi(t_2).$$

First, assume that $\chi(t_2)$ is nonnegative and set θ, M and $\bar{t}$ be defined by

$$\theta(t) = \chi(t) - \chi(t_1) - (t - t_1)\frac{\chi(t_2) - \chi(t_1)}{t_2 - t_1},$$
$$M = \text{Max}(\theta(t) : t \in [t_1, t_2]) \geq 0,$$

$$\bar{t} = \begin{cases} t_1 & \text{if } M = 0, \\ \text{Max}(t \in [t_1, t_2] : \theta(t) = M) & \text{otherwise.} \end{cases}$$

Then $t_1 \leq \bar{t} < t_2$ and for all $t \in (0, t_2 - t_1)$

$$\frac{\chi(\bar{t} + t) - \chi(\bar{t})}{t} \leq \frac{\chi(t_2) - \chi(t_1)}{t_2 - t_1} = \gamma < 0.$$

Note that $\chi(t_2) \geq 0$ implies $\chi(\bar{t}) > 0$. On the other hand, χ is continuous; hence for t positive small enough, $\chi(\bar{t} + t)$ is positive also.

Since ϕ is continuous we obtain by passing to the limit:

$$\frac{1}{\langle F(a + \bar{t}d), d\rangle} \liminf_{t \downarrow 0} \frac{1}{t} \ln \frac{\langle F(a + (t + \bar{t})d), d\rangle}{\langle F(a + \bar{t}d), d\rangle} < \lambda.$$

Assume now $\chi(t_2) < 0$. By continuity there exists $t_3 \in (t_1, t_2)$ such that $\chi(t_3) < 0$. Set now $d = -d, t_1 = -t_2, t_2 = -t_3$. Then we now have

$$t_1 < t_2 \quad \text{and} \quad \chi(t_1) > \chi(t_2) \geq 0.$$

We are just faced with the same situation as above. □

$m(F, a)$ is called the monotonicity index of F at a. The following theorem gives a more explicit expression when F is differentiable.

THEOREM 7.2. *Assume now that F is pseudomonotone and differentiable. Then*

$$m(F, a) = \inf_d \left(\frac{\langle F'(a)d, d\rangle}{\langle F(a), d\rangle^2}, \langle F(a), d\rangle \neq 0 \right).$$

Proof. The proof is an immediate consequence of Theorem 7.1. □

Note also that

$$m(F, a) = \inf_d (\langle F'(a)d, d\rangle : \langle F(a), d\rangle = 1),$$

and

$$m(F, a) = \inf_d \left(\left\langle \frac{1}{2}(F'(a) + F'(a)^t)d, d \right\rangle : \langle F(a), d\rangle = 1 \right).$$

When F is the gradient of a function f, then $m(F, a)$ coincides with $c(f, a)$ the convexity index of f at a introduced by Crouzeix and Lindberg, who give in [2, Proposition 10] a more explicit expression of $c(f, a)$. This expression remains true in the general case on the condition to replace $\nabla f(a)$ and $\nabla^2 f(a)$ by $F(a)$ and $\frac{1}{2}(F'(a) + F'(a)^t)$, respectively.

The same argument as in Crouzeix–Lindberg added to the first order characterization of pseudomonotone operators [8] gives the following theorem.

THEOREM 7.3. *$m(F,a)$ takes the following values.*

(i) *$+\infty$ if $F(a)=0$ and $\frac{1}{2}(F'(a)+F'(a)^t)$ is positive semi-definite (p.s.d.).*

(ii) *0 if $F(a)\neq 0$, $F'(a)+F'(a)^t$ is p.s.d. and the equation $\frac{1}{2}(F'(a)+F'(a)^t)\,v=F(a)$ has no solution.*

(iii) *$\frac{1}{\langle F(a),v\rangle}$ is $F(a)\neq 0\,v$ is a solution of the equation $\frac{1}{2}(F'(a)+F'(a)^t)\,v=F(a)$ and $F'(a)+F'(a)^t$ if p.s.d. or $((F'(a)+F'(a)^t)$ has one and only one negative eigenvalue and $\langle F(a),v\rangle<0)$.*

(iv) *$-\infty$ in all other cases.*

REFERENCES

[1] M. AVRIEL, *r-convex functions*, Math. Programming, 2 (1972), pp. 309–323.

[2] J.-P. CROUZEIX AND P. O. LINDBERG, *Additively decomposed quasiconvex functions*, Math. Programming, 35 (1986), pp. 42–57.

[3] G. DEBREU AND T. C. KOOPMANS, *Additively decomposed quasiconvex functions*, Math. Programming, 24 (1982), pp. 1–38.

[4] A. HASSOUNI, *Sous différentiels des fonctions quasiconvexes*, Doctorat de 3ème cycle de l'Université Paul Sabatier, Toulouse 3, 1983.

[5] ———, *Quasimonotone multifunctions: Applications to optimality conditions in quasiconvex programming*, Numer. Funct. Anal. Optim., 13 (1992), pp. 267–275.

[6] S. KARAMARDIAN, *Complementarity over cone with monotone and pseudomonotone maps*, J. Optim. Theory Appl., 18 (1976), pp. 167–181.

[7] S. KARAMARDIAN AND S. SCHAIBLE, *Seven kinds of monotone maps*, J. Optim. Theory Appl., 66 (1990), pp. 37–46.

[8] S. KARAMARDIAN, S. SCHAIBLE, AND J.-P. CROUZEIX, *Characterization of generalized monotone maps*, J. Optim. Theory Appl., 76 (1993), pp. 399–412.

[9] M. E. YAARI, *A note on separability and quasiconvexity*, Econometrica, 45 (1977), pp. 1183–1186.

SIAM J. OPTIMIZATION
Vol. 4, No. 3, pp. 659–675, August 1994
© 1994 Society for Industrial and Applied Mathematics
014

SENSITIVITY ANALYSIS OF OPTIMAL VALUE FUNCTIONS OF CONVEX PARAMETRIC PROGRAMS WITH POSSIBLY EMPTY SOLUTION SETS*

MOHAMMED MOUSSAOUI† AND ALBERTO SEEGER†

Abstract. Let $g(y_1, y_2)$ be the optimal value of the abstract program which consists in minimizing a convex function $f : X \to \mathbb{R} \cup \{+\infty\}$ over a feasible set of the form $\{x \in X : A_1x = y_1,\ A_2x \leq_K y_2\}$. Without assuming the existence of optimal solutions to this minimization problem, we derive formulas for the subdifferential and the approximate subdifferential of g. Several applications are discussed.

Key words. sensitivity analysis, convex programming, subdifferential, approximate subdifferential

AMS subject classifications. 90C31, 52A41

1. Introduction.

This paper is concerned with the sensitivity analysis of the optimal value $g(y_1, y_2)$ associated to the convex program

$$\underset{x \in X}{\text{Minimize}} \{f(x) : A_1x = y_1, A_2x \leq_K y_2\}. \tag{1.1}$$

Here $A_1x = y_1$ is a linear equality in the space Y_1, and $A_2x \leq_K y_2$ represents a linear inequality in the ordered space (Y_2, K). Sensitivity analysis of optimal value functions in convex parametric programming is a classic subject. The literature devoted to this topic is too rich to be quoted here. What makes our work original is that we do not assume the existence of optimal solutions to the problem (1.1). Convex parametric programs with empty solution sets are encountered quite often in practice. This fact is illustrated with the following specific example.

Example 1 (cf. Ekeland and Temam [7, §5]). Let $\Omega \subset \mathbb{R}^n$ be an open bounded regular set. For each $y \in W^{1,1}(\Omega)$ consider the minimal hypersurface problem

$$\begin{cases} \text{Minimize } \displaystyle\int_\Omega [1 + \|\text{grad}\, \varphi(\omega)\|^2]^{\frac{1}{2}}\, d\omega, \\ \varphi \in W^{1,1}(\Omega), \\ \varphi - y \in W_0^{1,1}(\Omega). \end{cases} \tag{1.2}$$

The above problem fits into the general scheme (1.1). Whether (1.2) has a solution or not depends on the choice of the "parameter" y. To be more specific, consider the crown $\Omega = \{\omega \in \mathbb{R}^2 : 1 < \|\omega\| < 2\}$ and a function y of the form $y(\omega) = c(2 - \|\omega\|)$. In this case (1.2) reduces to

$$\begin{cases} \text{Minimize } \displaystyle\int_1^2 \left[1 + \left(\frac{d\psi}{dr}\right)^2\right]^{\frac{1}{2}} r\, dr, \\ \psi(1) = c, \\ \psi(2) = 0. \end{cases} \tag{1.3}$$

As pointed out in [7, p. 127], this minimization problem has a solution only if $c \in \mathbb{R}$ lies below a certain threshold value.

The main concern of this work is to derive formulas for the subdifferential $\partial g(\bar{y}_1, \bar{y}_2)$ and the approximate subdifferential $\partial_\varepsilon g(\bar{y}_1, \bar{y}_2)$. These mathematical objects contain first- and higher-order information on the behavior of g around the given point $(\bar{y}_1, \bar{y}_2)$.

* Received by the editors June 3, 1992; accepted for publication (in revised form) November 24, 1993.
† University of Avignon, Department of Mathematics, 33 rue Louis Pasteur, 84000 Avignon, France.

The paper is organized as follows. In §2 we study in detail the purely equality-constrained case. In the first part of §3 we discuss the purely inequality-constrained case. At the end of §3 we consider the general problem (1.1). Section 4 is left to the discussion of several applications.

2. Convex programming under linear equality constraints. This section deals with the sensitivity analysis of the optimal value function associated to the linearly constrained convex minimization problem

$$(P)_y \qquad \begin{cases} \text{Minimize } f(x), \\ Ax = y. \end{cases}$$

To be more precise, we study the behavior of the optimal value function

$$y \in Y \mapsto f_A(y) := \inf_{Ax=y} f(x) \tag{2.1}$$

around a given point $\bar{y} \in Y$. Thus, y is interpreted as a parameter vector which has been perturbed from its reference level $\bar{y}$. The function f_A is sometimes referred to as the image of f under A (cf. [22, p. 38]), or as the Schur complement of f relative to A (cf. [27, p. 52]).

The parameter space Y is assumed to be a locally convex real topological linear space. A similar assumption is made on the space X of decision variables. Throughout this section one supposes that

$$\begin{cases} f : X \to \mathbb{R} \cup \{+\infty\} & \text{is a convex proper function;} \\ A : X \to Y & \text{is a continuous linear operator;} \\ f_A & \text{is finite at } \bar{y} \in Y. \end{cases} \tag{2.2}$$

Under the above assumptions, the optimal value function f_A is convex but not necessarily differentiable. A concept which reflects the first-order behavior of f_A around $\bar{y}$ is the set

$$\partial f_A(\bar{y}) := \{v \in Y^* : f_A(y) \geq f_A(\bar{y}) + \langle v, y - \bar{y} \rangle \text{ for all } y \in Y\}. \tag{2.3}$$

This set is called the *subdifferential* of f_A at $\bar{y}$ (cf. [13, p. 46]). Here Y^* is another locally convex topological linear space which has been paired with Y by means of a separating bilinear form $\langle .,. \rangle : Y^* \times Y \to \mathbb{R}$. The topologies on Y and Y^* are supposed to be compatible with respect to this pairing (see Rockafellar [21, §3] for details). Additional information on the behavior of f_A can be captured by introducing a parameter $\varepsilon > 0$ and enlarging the set $\partial f_A(\bar{y})$ in the following manner:

$$\partial_\varepsilon f_A(\bar{y}) := \{v \in Y^* : f_A(y) \geq f_A(\bar{y}) + \langle v, y - \bar{y} \rangle - \varepsilon \text{ for all } y \in Y\}. \tag{2.4}$$

In Hiriart-Urruty and Seeger [12] and Seeger [26], [28], it is explained how this set can be used to obtain, for instance, second-order information on f_A around $\bar{y}$. The set (2.4) is usually referred to as the *approximate subdifferential* of f_A at $\bar{y}$. Approximate subdifferentials have many other uses in convex optimization: design of algorithms [6], [14], [32], characterization of approximate optimal solutions [8], [19], and formulation of variational principles [3], [4] are just a few examples. The (exact) subdifferential (2.3) corresponds, of course, to the case $\varepsilon = 0$.

The approximate subdifferential of the optimal value function f_A can be expressed in terms of the approximate subdifferential of the objective function f. The next calculus rule is known and recorded here just for the sake of completeness in our exposition. It involves the set of optimal solutions

$$S(\bar{y}) := \{x \in X : Ax = \bar{y},\ f(x) = f_A(\bar{y})\} \tag{2.5}$$

to the unperturbed problem $(P)_{\bar{y}}$, and the approximate subdifferential

$$\partial_\varepsilon f(\bar{x}) = \{u \in X^* : f(x) \geq f(\bar{x}) + \langle u, x - \bar{x}\rangle - \varepsilon \text{ for all } x \in X\}$$

of f at a point $\bar{x}$ in $S(\bar{y})$. In the above line the symbol $\langle .,. \rangle$ denotes this time a separating bilinear form pairing the spaces X and X^*.

PROPOSITION 1. *With the general assumption* (2.2), *let* $\varepsilon \geq 0$ *and* $\bar{x}$ *be any element in the solution set* $S(\bar{y})$. *Then*

$$\partial_\varepsilon f_A(\bar{y}) = \{v \in Y^* : A^*v \in \partial_\varepsilon f(\bar{x})\}, \tag{2.6}$$

where $A^* : Y^* \to X^*$ *stands for the adjoint operator of* $A : X \to Y$.

The above result can be found, for instance, in Hiriart-Urruty [9, p. 74] or Zalinescu [31]. The writing of formula (2.6) makes sense only if such an element $\bar{x}$ does exist. The nonemptiness assumption on the solution set $S(\bar{y})$ is the main drawback of Proposition 1. To circumvent the troubles that might arise due to the lack of optimal solutions, we introduce a parameter $\eta > 0$ and enlarge the set $S(\bar{y})$ in the following way:

$$S_\eta(\bar{y}) := \{x \in X : Ax = \bar{y},\ f(x) \leq f_A(\bar{y}) + \eta\}. \tag{2.7}$$

The elements of $S_\eta(\bar{y})$ are interpreted as optimal solutions to $(P)_{\bar{y}}$ within a tolerance η. In short, these elements are *approximate solutions* to $(P)_{\bar{y}}$.

Now we are ready to state one of the main results of this paper.

THEOREM 1. *With the general assumption* (2.2), *let the set of approximate solutions* $S_\eta(\bar{y})$ *be defined by* (2.7). *Then, for all* $\varepsilon \geq 0$, *one has*

$$\begin{aligned}\partial_\varepsilon f_A(\bar{y}) &= \bigcap_{\eta>0} \bigcap_{x \in S_\eta(\bar{y})} \{v \in Y^* : A^*v \in \partial_{\varepsilon+\eta} f(x)\} \\ &= \bigcap_{\eta>0} \bigcup_{Ax=\bar{y}} \{v \in Y^* : A^*v \in \partial_{\varepsilon+\eta} f(x)\}.\end{aligned}$$

Proof. Clearly the set

$$M_\eta(\bar{y}) := \bigcap_{x \in S_\eta(\bar{y})} \{v \in Y^* : A^*v \in \partial_{\varepsilon+\eta} f(x)\} \tag{2.8}$$

is contained in

$$N_\eta(\bar{y}) := \bigcup_{Ax=\bar{y}} \{v \in Y^* : A^*v \in \partial_{\varepsilon+\eta} f(x)\}. \tag{2.9}$$

It suffices then to prove the inclusions

$$\partial_\varepsilon f_A(\bar{y}) \subset \bigcap_{\eta>0} M_\eta(\bar{y}), \tag{2.10}$$

$$\bigcap_{\eta>0} N_\eta(\bar{y}) \subset \partial_\varepsilon f_A(\bar{y}). \tag{2.11}$$

To prove the first one, choose arbitrarily $\bar{v} \in \partial_\varepsilon f_A(\bar{y})$, $\eta > 0$, and $x \in S_\eta(\bar{y})$. The condition $\bar{v} \in \partial_\varepsilon f_A(\bar{y})$ can be written in the form

$$(f_A)^*(\bar{v}) + f_A(\bar{y}) - \langle \bar{v}, \bar{y}\rangle \leq \varepsilon,$$

where

$$v \in Y^* \mapsto (f_A)^*(v) := \operatorname*{Sup}_{y \in Y}\{\langle v, y\rangle - f_A(y)\}$$

stands for the Fenchel conjugate of f_A. By combining the equality $Ax = \bar{y}$ and the general formula (cf. [13, p. 179])

$$(f_A)^* = f^* \circ A^*, \tag{2.12}$$

one gets

$$f^*(A^*\bar{v}) + f_A(\bar{y}) - \langle A^*\bar{v}, x\rangle \leq \varepsilon.$$

Since $f(x) \leq f_A(\bar{y}) + \eta$, one can write finally

$$f^*(A^*\bar{v}) + f(x) - \langle A^*\bar{v}, x\rangle \leq \varepsilon + \eta.$$

This means that $A^*\bar{v} \in \partial_{\varepsilon+\eta} f(x)$, and thus the proof of (2.10) is complete. Now take an arbitrary $\bar{v}$ in the intersection of the family $\{N_\eta(\bar{y}) : \eta > 0\}$. For any $\eta > 0$ there exists an element $x \in X$ such that

$$Ax = \bar{y}, \qquad A^*\bar{v} \in \partial_{\varepsilon+\eta} f(x),$$

and, a posteriori,

$$f^*(A^*\bar{v}) + f(x) - \langle \bar{v}, \bar{y}\rangle \leq \varepsilon + \eta.$$

Again using the general formula (2.12) and the fact that $f_A(\bar{y}) \leq f(x)$, one obtains

$$(f_A)^*(\bar{v}) + f_A(\bar{y}) - \langle \bar{v}, \bar{y}\rangle \leq \varepsilon + \eta.$$

Since the above inequality is true for any $\eta > 0$, we get finally $\bar{v} \in \partial_\varepsilon f_A(\bar{y})$. This completes the proof of inclusion (2.11). □

Remark 1. Theorem 1 remains true even without assuming the convexity of f. However, in the nonconvex case the set $\partial_\varepsilon f_A(\bar{y})$ does not necessarily reflect the behavior of f_A around the point $\bar{y}$. For this reason we will always work in a convex setting.

Remark 2. Theorem 1 is a generalization of Proposition 1. It can be shown that if the solution set $S(\bar{y})$ happens to be nonempty, then both formulas for $\partial_\varepsilon f_A(\bar{y})$ given in Theorem 1 reduce to the expression appearing on the right-hand side of (2.6).

The general Schur complementation theory for multivalued mappings provides a framework for the understanding and for a compact writing of the second formula appearing in Theorem 1.

DEFINITION 1. *The Schur complement of the multivalued mapping* $\Gamma : X \to X^*$, *relative to the continuous linear operator* $A : X \to Y$, *is the multivalued mapping* $[\Gamma]_A : Y \to Y^*$ *given by*

$$[\Gamma]_A(y) := \bigcup_{Ax=y} \{v \in Y^* : A^*v \in \Gamma(x)\},$$

or equivalently,

$$[\Gamma]_A(y) := \{-v \in Y^* : \textit{for some } x \in X, \textit{ the pair } (x, v) \textit{ solves the system } (2.13)\},$$

where

$$\begin{cases} 0 \in \Gamma(x) + A^*v, \\ y = Ax. \end{cases} \tag{2.13}$$

The first part of (2.13) can be viewed as a stationarity (or equilibrium) condition associated with a variational problem. The second part of (2.13) usually corresponds to a constitutive law of the system under consideration (for instance, Kirchhoff's conservation law in an electrical network, or Hooke's elasticity law in a mechanical system). The term $v \in Y^*$ is interpreted as a dual variable, or as a vector of Lagrange multipliers. To fix the ideas, take Γ as the subdifferential mapping $\partial f : X \to X^*$ of the convex proper function $f : X \to \mathbb{R} \cup \{+\infty\}$. The pair $(\bar{x}, \bar{v})$ solves the system

$$0 \in \partial f(x) + A^*v, \qquad \bar{y} = Ax \tag{2.14}$$

if and only if $(\bar{x}, \bar{v})$ is a saddle point of the Lagrangian

$$(x, v) \in X \times Y^* \mapsto L_{\bar{y}}(x, v) = f(x) + \langle v, Ax - \bar{y} \rangle. \tag{2.15}$$

In this case the first component $\bar{x}$ is an optimal solution to the primal problem $(P)_{\bar{y}}$. In particular, if $(P)_{\bar{y}}$ has no optimal solutions, then $[\partial f]_A$ is empty valued at $\bar{y}$. The vacuity of $[\Gamma]_A$ at $\bar{y}$ can be avoided by passing from $\Gamma = \partial f$ to an enlargement $\Gamma = \partial_\eta f$, with $\eta > 0$. Enlarged or relaxed systems of the form

$$0 \in \partial_\eta f(x) + A^*v, \qquad \bar{y} = Ax$$

have been already considered in a different context by Heukemes, Nguyen, and Strodiot [8] and Robinson [20]. An extension to the framework of relaxed saddle points of general convex-concave Lagrangians can be found in McLinden [15], [16].

Without further ado we state below an alternative writing for the second formula appearing in Theorem 1.

COROLLARY 1. *Consider the general assumption* (2.2). *Then, for all* $\varepsilon \geq 0$, *one has*

$$\partial_\varepsilon f_A(\bar{y}) = \bigcap_{\eta > 0} [\partial_{\varepsilon+\eta} f]_A(\bar{y}). \tag{2.16}$$

Proof. Consider again the set $N_\eta(\bar{y})$ introduced in (2.9). According to Theorem 1 this set is precisely the multivalued mapping $[\partial_{\varepsilon+\eta} f]_A$ evaluated at $\bar{y}$. □

We end this section by giving another formula for $\partial_\varepsilon f_A(\bar{y})$ which applies only to the case $\varepsilon > 0$. The next corollary says that up to a closure operation, the approximate subdifferential mapping $\partial_\varepsilon f_A$ coincides with $[\partial_\varepsilon f]_A$.

COROLLARY 2. *Besides the general assumption* (2.2), *suppose also that* f_A *is lower-semicontinuous at* $\bar{y}$, *and minorized by some continuous affine function. Then, for all* $\varepsilon > 0$, *one has*

$$\partial_\varepsilon f_A(\bar{y}) = \overline{\bigcup_{Ax = \bar{y}} \{v \in Y^* : A^*v \in \partial_\varepsilon f(x)\}}. \tag{2.17}$$

Proof. The second formula in Theorem 1 implies that the set

$$[\partial_\varepsilon f]_A(\bar{y}) = \bigcup_{Ax = \bar{y}} \{v \in Y^* : A^*v \in \partial_\varepsilon f(x)\}$$

is contained in $\partial_\varepsilon f_A(\bar{y})$. Thus,

$$\overline{[\partial_\varepsilon f]_A(\bar{y})} \subset \partial_\varepsilon f_A(\bar{y}).$$

The above closure operation has been taken, of course, with respect to the topology in Y^*. The proof of the reverse inclusion is more delicate. Let us prove first the equality

$$[\partial_\varepsilon f]_A(\bar{y}) = \bigcup \{\partial_{\alpha(x)} f_A(\bar{y}) : Ax = \bar{y}, \quad \alpha(x) \geq 0\}. \tag{2.18}$$

Here we use the notation

$$\alpha(x) := \varepsilon + f_A(\bar{y}) - f(x),$$

noting that $\varepsilon > 0$ and $\bar{y}$ are fixed. For an arbitrary point $x \in X$ satisfying $Ax = \bar{y}$ and $\alpha(x) \geq 0$, the following conditions are equivalent:

(i) $A^*v \in \partial_\varepsilon f(x)$;
(ii) $f(x') \geq f(x) + \langle A^*v, x' - x\rangle - \varepsilon$ for all $x' \in X$;
(iii) $f_A(y) \geq f(x) + \langle v, y - \bar{y}\rangle - \varepsilon$ for all $y \in Y$;
(iv) $v \in \partial_{\alpha(x)} f_A(\bar{y})$.

This, of course, completes the proof of equality (2.18). Denote by Ω the set on the right-hand side of (2.18). It suffices now to prove that the closure of this set coincides with $\partial_\varepsilon f_A(\bar{y})$. It is a simple matter to check that

$$\Omega = \begin{cases} \partial_\varepsilon f_A(\bar{y}) & \text{if } S(\bar{y}) \text{ is nonempty,} \\ \bigcup\limits_{0 \leq \alpha < \varepsilon} \partial_\alpha f_A(\bar{y}) & \text{if } S(\bar{y}) \text{ is empty.} \end{cases}$$

If $S(\bar{y})$ is nonempty, then there is nothing to prove. Observe incidentally that in this particular case the closure operation in (2.17) becomes superfluous. If $S(\bar{y})$ is empty, then we invoke a general property of the approximate subdifferential mapping $\alpha \mapsto \partial_\alpha f_A(\bar{y})$, namely,

$$\overline{\bigcup_{0 \leq \alpha < \varepsilon} \partial_\alpha f_A(\bar{y})} = \partial_\varepsilon f_A(\bar{y}). \tag{2.19}$$

The above equality can be proven in an easy way. Observe first that

$$\bigcup_{0 \leq \alpha < \varepsilon} \partial_\alpha f_A(\bar{y}) = \{v \in Y^* : r(v) < \varepsilon\},$$

where

$$v \in Y^* \mapsto r(v) := (f_A)^*(v) + (f_A)(\bar{y}) - \langle v, \bar{y}\rangle$$

is a convex lower-semicontinuous function whose infimum is equal to 0. Since ε is strictly greater than this infimum, one can write

$$\overline{\{v \in Y^* : r(v) < \varepsilon\}} = \{v \in Y^* : r(v) \leq \varepsilon\}.$$

This completes the proof of (2.19), and also of the corollary. □

Let us illustrate Corollary 2 with the help of a simple example.

Example 2. Let $f : \mathbb{R}^2 \to \mathbb{R}$ and $A : \mathbb{R}^2 \to \mathbb{R}$ be defined by $f(x_1, x_2) = [\exp(2x_1) + \frac{1}{4}(x_2)^4]^{1/2}$ and $A(x_1, x_2) = x_2$, respectively. Then $f_A : \mathbb{R} \to \mathbb{R}$ is given by $f_A(y) = \frac{1}{2}y^2$.

For $\bar{y} = 0$ and $\varepsilon > 0$, the open interval $[\partial_\varepsilon f]_A(\bar{y}) =]-\sqrt{2\varepsilon}, \sqrt{2\varepsilon}[$ is strictly contained in $\partial_\varepsilon f_A(\bar{y}) = [-\sqrt{2\varepsilon}, \sqrt{2\varepsilon}]$. This example shows that the closure operation in (2.17) cannot be omitted. Now consider the case $\varepsilon = 0$. Clearly, the set $[\partial f]_A(\bar{y})$ is empty and $\partial f_A(\bar{y}) = \{0\}$. So, this same example shows that formula (2.17) does not apply for the exact subdifferential.

Remark 3. General results guaranteeing the lower-semicontinuity of f_A can be found, for instance, in McLinden [17] and Zalinescu [31]. If f_A is not necessarily lower-semicontinuous at $\bar{y}$, then formula (2.17) remains true for all $\varepsilon > f_A(\bar{y}) - (cl\, f_A)(\bar{y})$. The notation $cl\, f_A$ stands for the lower-semicontinuous hull of f_A. The case $\varepsilon < f_A(\bar{y}) - (cl\, f_A)(\bar{y})$ has no interest since it implies that $\partial_\varepsilon f_A(\bar{y})$ is empty.

3. Convex programs with linear equality and inequality constraints. Consider now the sensitivity analysis of the optimal value function associated to the inequality-constrained convex minimization program

$$(\tilde{P})_y \qquad \begin{cases} \text{Minimize } f(x), \\ Ax \leq_K y. \end{cases}$$

Here the inequality $Ax \leq_K y$ means that $y - Ax \in K$. In fact, linear equality and inequality constraints can both be incorporated into the "cone constraint" $y - Ax \in K$. For this reason we prefer to keep the notation in this more abstract level, and we write accordingly

$$f_{A,K}(y) := \operatorname*{Inf}_{x \in X} \{f(x) : y - Ax \in K\}. \tag{3.1}$$

To remain within the realm of convex analysis and to avoid trivial situations, we suppose the following set of hypotheses to be true:

$$\begin{cases} f : X \to \mathbb{R} \cup \{+\infty\} & \text{is a convex proper function;} \\ A : X \to Y & \text{is a continuous linear operator;} \\ K & \text{is a closed convex cone in } Y; \\ f_{A,K} & \text{is finite at } \bar{y} \in Y. \end{cases} \tag{3.2}$$

The Lagrangian associated with the problem $(\tilde{P})_{\bar{y}}$ takes the same form as in (2.15), namely,

$$L_{\bar{y}}(x, v) = f(x) + \langle v, Ax - \bar{y} \rangle.$$

This time, however, only "nonnegative" multiplier vectors are of interest, in the sense that we invariably select v in the set

$$K^+ := \{v \in Y^* : \langle v, k \rangle \geq 0 \text{ for all } k \in K\}.$$

The above closed convex cone is known as the positive polar of K. It helps to see v also as the decision variable in the dual problem associated with $(\tilde{P})_{\bar{y}}$, namely,

$$(\tilde{D})_{\bar{y}} \qquad \begin{cases} \text{Maximize } \{\langle -v, \bar{y} \rangle - f^*(-A^*v)\}, \\ v \in K^+. \end{cases}$$

The classical theory of convex programming tells us that $(\bar{x}, \bar{v})$ is a saddle point of $L_{\bar{y}}(.,.)$ over $X \times K^+$ if and only if $(\bar{x}, \bar{v})$ solves the system

$$(S)_{\bar{y}} \qquad \begin{cases} 0 \in \partial f(x) + A^*v & \text{(stationarity)}, \\ \bar{y} - Ax \in K & \text{(primal feasibility)}, \\ v \in K^+ & \text{(dual feasibility)}, \\ \langle v, \bar{y} - Ax \rangle = 0 & \text{(complementary slackness)}. \end{cases}$$

In this case $\bar{x}$ is an optimal solution to the primal problem $(\tilde{P})_{\bar{y}}$, and $\bar{v}$ is an optimal solution to the dual problem $(\tilde{D})_{\bar{y}}$. The troubles that might arise due to the lack of optimal solutions to $(\tilde{P})_{\bar{y}}$, or due to the unsolvability of $(S)_{\bar{y}}$, can be circumvented by considering a *relaxed system* of the form

$$(S)_{\bar{y}}^{\varepsilon_1,\varepsilon_2} \quad \begin{cases} 0 \in \partial_{\varepsilon_1} f(x) + A^* v & (\varepsilon_1\text{-stationarity}), \\ \bar{y} - Ax \in K & (\text{primal feasibility}), \\ v \in K^+ & (\text{dual feasibility}), \\ \langle v, \bar{y} - Ax \rangle \leq \varepsilon_2 & (\varepsilon_2\text{-complementary slackness}). \end{cases}$$

Here ε_1 and ε_2 are two nonnegative numbers that are supposed to sum up to a prescribed positive value. In terms of this relaxed system one can write a compact formula for the approximate subdifferential of $f_{A,K}$.

THEOREM 2. *Consider the general assumption* (3.2). *Then, for all* $\varepsilon \geq 0$, *one has*

$$\partial_\varepsilon f_{A,K}(\bar{y}) = \bigcap_{\eta>0} \bigcup_{x\in X} \bigsqcup_{\substack{\varepsilon_1\geq 0,\varepsilon_2\geq 0 \\ \varepsilon_1+\varepsilon_2=\varepsilon+\eta}} \{-v \in Y^* : (x,v) \text{ solves } (S)_{\bar{y}}^{\varepsilon_1,\varepsilon_2}\}. \tag{3.3}$$

Proof. Write $f_{A,K}$ in the form

$$f_{A,K}(y) = \operatorname*{Inf}_{x,s}\{f(x) : Ax + s = y,\ s \in K\},$$

where $s \in Y$ is interpreted as a vector of "slack variables." The restriction $s \in K$ can be incorporated into the objective function, and so we can write

$$f_{A,K}(y) = \operatorname*{Inf}_{B(x,s)=y} (f \oplus \Psi_K)(x,s). \tag{3.4}$$

Here $B : X \times Y \to Y$ and $f \oplus \Psi_K : X \times Y \to \mathbb{R} \cup \{+\infty\}$ are given, respectively, by

$$B(x,s) = Ax + s, \qquad (f \oplus \Psi_K)(x,s) = f(x) + \Psi_K(s),$$

and Ψ_K denotes the indicator function of K (i.e., $\Psi_K(s) = 0$ if $s \in K$, $+\infty$ if not). Now we apply Theorem 1 to the optimal value function $f_{A,K}$ as given by (3.4). In this way we obtain

$$\partial_\varepsilon f_{A,K}(\bar{y}) = \bigcap_{\eta>0} \bigcup_{B(x,s)=\bar{y}} \{v \in Y^* : B^* v \in \partial_{\varepsilon+\eta}(f \oplus \Psi_K)(x,s)\}. \tag{3.5}$$

To evaluate the set appearing on the right-hand side of (3.5), notice first that the adjoint $B^* : Y^* \to X^* \times Y^*$ of B is given by $B^* v = (A^* v, v)$. As a matter of direct computation one also obtains

$$\partial_{\varepsilon+\eta}(f \oplus \Psi_K)(x,s) = \bigsqcup_{\substack{\varepsilon_1\geq 0,\varepsilon_2\geq 0 \\ \varepsilon_1+\varepsilon_2=\varepsilon+\eta}} \{\partial_{\varepsilon_1} f(x) \times \partial_{\varepsilon_2} \Psi_K(s)\}.$$

According to Hiriart-Urruty [11, p. 222] the approximate subdifferential $\partial_{\varepsilon_2} \Psi_K(s)$, which represents geometrically the set of ε_2-normal directions to K at s, is given by

$$\partial_{\varepsilon_2} \Psi_K(s) = \begin{cases} \{-v : v \in K^+ : \langle v, s \rangle \leq \varepsilon_2\} & \text{if } s \in K, \\ \text{empty set} & \text{if } s \notin K. \end{cases}$$

To complete the proof it suffices to get rid of the slack variable s (just write $\bar{y} - Ax$ instead of s), and to then join all the pieces together. □

Remark 4. Theorem 2 extends in several ways a result due to Shiraishi [30, Thm. 3.1]. First, we do not assume the existence of optimal solutions to the unperturbed problem $(\tilde{P})_{\bar{y}}$. Second, Shiraishi works in a finite-dimensional setting, say $X = \mathbb{R}^n$ and $Y = \mathbb{R}^m$, and he chooses K as the usual Pareto ordering cone $K = \mathbb{R}^m_+$. Third, Shiraishi assumes an additional constraint qualification hypothesis, namely, the Slater condition : $A\tilde{x} \in \operatorname{int} \mathbb{R}^m_+$ for some $\tilde{x} \in \mathbb{R}^n$.

The writing of formula (3.3) is considerably simplified in the case $\varepsilon = 0$. This particular case deserves a special mention.

COROLLARY 3. *Under the general assumption* (3.2), *the subdifferential of* $f_{A,K}$ *at* $\bar{y}$ *is given by*

$$\partial f_{A,K}(\bar{y}) = \bigcap_{\eta>0} \bigcup_{Ax \leq_K \bar{y}} \{v \in K^- : A^*v \in \partial_\eta f(x),\ \langle v, Ax - \bar{y}\rangle \leq \eta\}, \tag{3.6}$$

where $K^- := -K^+$ *stands for the negative polar of* K.

Proof. Consider again the equality (3.5) given in the proof of Theorem 2. For $\varepsilon = 0$ it takes the form

$$\partial f_{A,K}(\bar{y}) = \bigcap_{\eta>0} \bigcup_{B(x,s)=\bar{y}} \{v \in Y^* : B^*v \in \partial_\eta (f \oplus \Psi_K)(x,s)\},$$

with

$$\partial_\eta (f \oplus \Psi_K)(x,s) = \bigcup_{\substack{\varepsilon_1 \geq 0, \varepsilon_2 \geq 0 \\ \varepsilon_1 + \varepsilon_2 = \eta}} \{\partial_{\varepsilon_1} f(x) \times \partial_{\varepsilon_2} \Psi_K(s)\}.$$

From the above line one obtains the bounds

$$\partial_{\eta/2} f(x) \times \partial_{\eta/2} \Psi_K(s) \subset \partial_\eta (f \oplus \Psi_K)(x,s) \subset \partial_\eta f(x) \times \partial_\eta \Psi_K(s),$$

and consequently

$$\partial f_{A,K}(\bar{y}) = \bigcap_{\eta>0} \bigcup_{B(x,s)=\bar{y}} \{v \in Y^* : B^*v \in \partial_\eta f(x) \times \partial_\eta \Psi_K(s)\}.$$

The remaining part follows the same steps as in the proof of Theorem 2. □

Finally, we consider the sensitivity analysis of the optimal value function

$$(y_1, y_2) \in Y_1 \times Y_2 \mapsto g(y_1, y_2) = \operatorname*{Inf}_{x \in X} \{f(x) : A_1 x = y_1,\ A_2 x \leq_K y_2\}$$

associated with the general convex parametric program (1.1). Interesting examples of minimization problems that fit into this scheme abound in the literature.

Our basic set of assumptions is as follows:

$$\begin{cases} f : X \to \mathbb{R} \cup \{+\infty\} & \text{is a convex proper function;} \\ A_1 : X \to Y_1 \quad \text{and} \quad A_2 : X \to Y_2 & \text{are continuous linear operators;} \\ K & \text{is a closed convex cone in } Y_2; \\ g & \text{is finite at } (\bar{y}_1, \bar{y}_2) \in Y_1 \times Y_2. \end{cases} \tag{3.7}$$

As we have done before, we continue to work in the context of paired locally convex real topological linear spaces.

A formula for the approximate subdifferential of g is given in the theorem below.

THEOREM 3. *Consider the general assumption* (3.7). *Then, for all* $\varepsilon \geq 0$, *one has*

(3.8)

$$\partial_\varepsilon g(\bar{y}_1, \bar{y}_2) = \bigcap_{\eta>0} \bigcup_{x\in X} \bigsqcup_{\substack{\varepsilon_1\geq 0, \varepsilon_2 \geq 0 \\ \varepsilon_1+\varepsilon_2=\varepsilon+\eta}} \{-(v_1, v_2) \in Y_1^* \times Y_2^* : (x, v_1, v_2) \text{ solves } (S)^{\varepsilon_1,\varepsilon_2}_{\bar{y}_1,\bar{y}_2}\},$$

where

$$(S)^{\varepsilon_1,\varepsilon_2}_{\bar{y}_1,\bar{y}_2} \quad \begin{cases} 0 \in \partial_{\varepsilon_1} f(x) + A_1^* v_1 + A_2^* v_2 & (\varepsilon_1\text{-stationarity}), \\ A_1 x = \bar{y}_1, \quad A_2 x \leq_K \bar{y}_2 & (\text{primal feasibility}), \\ v_1 \in Y_1^*, \quad v_2 \in K^+ & (\text{dual feasibility}) \\ \langle v_2, \bar{y}_2 - A_2 x\rangle \leq \varepsilon_2 & (\varepsilon_2\text{-complementary slackness}). \end{cases}$$

Proof. Write g in the form

$$g(y_1, y_2) = \operatorname*{Inf}_{x\in X} \{f(x) : (y_1, y_2) - (A_1 x, A_2 x) \in \{0_{Y_1}\} \times K\},$$

where 0_{Y_1} stands for the origin in Y_1. Observe that $\{0_{Y_1}\} \times K$ is a closed convex cone in the product space $Y_1 \times Y_2$, so we are in the context of an optimal value function of the form (3.1). Theorem 2 does the rest of the job. □

We close this section by working out formula (3.8) for the particular case $\varepsilon = 0$. The next result is proven in the same way as Corollary 3.

COROLLARY 4. *Under the general assumption* (3.7), *the subdifferential of* g *at* $(\bar{y}_1, \bar{y}_2)$ *is given by*

$$\partial g(\bar{y}_1, \bar{y}_2) = \bigcap_{\eta>0} \bigsqcup_{\substack{A_1 x=\bar{y}_1 \\ A_2 x\leq_K \bar{y}_2}} \{(v_1, v_2) \in Y_1^* \times K^- : A_1^* v_1 + A_2^* v_2 \in \partial_\eta f(x), \langle v_2, A_2 x - \bar{y}_2\rangle \leq \eta\}.$$

4. Applications.

4.1. Marginal functions and parametric problems of the calculus of variations. Let f be defined over the product space $X \times Y$. Here X and Y are two locally convex real topological linear spaces paired with X^* and Y^*, respectively. The product space $X \times Y$ is paired with $X^* \times Y^*$ in the usual way, that is, by means of the separating bilinear form

$$\langle (x^*, y^*), (x, y)\rangle = \langle x^*, x\rangle + \langle y^*, y\rangle.$$

The *marginal function* m is defined on Y by

$$m(y) = \operatorname*{Inf}_{x\in X} f(x, y). \tag{4.1}$$

Formulas for the exact and approximate subdifferentials of m can be obtained by using the results established in §2.

COROLLARY 5. *Let* $f : X \times Y \to \mathbb{R} \cup \{+\infty\}$ *be a convex proper function. Suppose the marginal function* m *is finite at* $\bar{y} \in Y$, *and denote by*

$$S_\eta(\bar{y}) := \{x \in X : f(x, \bar{y}) \leq m(\bar{y}) + \eta\} \tag{4.2}$$

the set of elements that minimize the function $f(.,\bar{y})$ within a tolerance $\eta > 0$. Then, for all $\varepsilon \geq 0$, we have

$$\partial_\varepsilon m(\bar{y}) = \bigcap_{\eta>0} \bigcap_{x\in S_\eta(\bar{y})} \{v \in Y^* : (0,v) \in \partial_{\varepsilon+\eta} f(x,\bar{y})\}$$
$$= \bigcap_{\eta>0} \bigcup_{x\in X} \{v \in Y^* : (0,v) \in \partial_{\varepsilon+\eta} f(x,\bar{y})\}.$$

Proof. It suffices to apply Theorem 1 to the case in which $A : X \times Y \to Y$ is given by $A(x,y) = y$. The adjoint operator $A^* : Y^* \to X^* \times Y^*$ takes the form $A^*v = (0,v)$ here. □

For the sake of completeness we also record another formula for $\partial_\varepsilon m(\bar{y})$. It is obtained as an immediate consequence of Corollary 2 and applies only to the case $\varepsilon > 0$.

COROLLARY 6. *Let $f : X \times Y \to R \cup \{+\infty\}$ be a convex proper function. Suppose the marginal function m is finite and lower-semicontinuous at $\bar{y} \in Y$. Suppose also that m is minorized by some continuous affine function. Then, for all $\varepsilon > 0$, one has*

$$\partial_\varepsilon m(\bar{y}) = \overline{\bigcup_{x\in X} \{v \in Y^* : (0,v) \in \partial_\varepsilon f(x,\bar{y})\}}. \tag{4.3}$$

According to Remark 2, if there exists an (exact) minimizer of $f(.,\bar{y})$, i.e., if the set

$$S(\bar{y}) := \{x \in X : f(x,\bar{y}) = m(\bar{y})\}$$

is nonempty, then the expression for $\partial_\varepsilon m(\bar{y})$ takes the simpler form

$$\partial_\varepsilon m(\bar{y}) = \{v \in Y^* : (0,v) \in \partial_\varepsilon f(\bar{x},\bar{y})\}, \tag{4.4}$$

where $\bar{x}$ is an arbitrary element in $S(\bar{y})$. Formula (4.4) is known and can be found, for instance, in Hiriart-Urruty [9, p. 73].

To illustrate Corollary 5, let us study the dependence on the parameter $y \in L^2[0,1]$ of the optimal value $m(y)$ associated with the convex program

$$\begin{cases} \text{Minimize } \displaystyle\int_0^1 t^2[y(t)^2 + [\dot{x}(t)]^2]^{\frac{1}{2}}\, dt, \\ x \in W^{1,2}[0,1], \\ x(0) = 0, \qquad x(1) = 1. \end{cases} \tag{4.5}$$

The sensitivity analysis of a convex parametric problem of the calculus of variations written in its full generality is the subject of a paper [29] by Seeger. Trying to describe the details here would carry us too far from our main topic. The particular problem (4.5) is interesting enough and serves to illustrate our results. So, we are concerned with a particular problem of the calculus of variations with fixed endpoints and trajectories lying in the space

$$W^{1,2}[0,1] := \{x : [0,1] \to \mathbb{R} : x \text{ is absolutely continuous and } \dot{x} \in L^2[0,1]\}.$$

If the parameter y is set at the reference level

$$\bar{y}(t) = 0 \quad \text{for a.e. } t \in [0,1],$$

one obtains the unperturbed problem

$$\text{(4.6)} \qquad \begin{cases} \text{Minimize } \displaystyle\int_0^1 t^2|\dot{x}(t)|dt, \\ x \in W^{1,2}[0,1], \\ x(0) = 0, \qquad x(1) = 1. \end{cases}$$

This is a classic example of a nonsmooth convex variational calculus problem which has no optimal solution. The optimal value of (4.6) is equal to 0. To see this, just consider the minimizing sequence

$$x_k(t) = \begin{cases} kt & \text{if } 0 \le t \le \dfrac{1}{k}, \\ 1 & \text{if } \dfrac{1}{k} \le t \le 1. \end{cases}$$

The optimal value 0 is, however, not attained.

To study the sensitivity of the function m, it helps to see (4.5) as a problem of optimal control. Introduce, then, the control variable $u \in L^2[0,1]$ related to the trajectory $x \in W^{1,2}[0,1]$ by means of the state equation $\dot{x}(t) = u(t)$. In this way one transforms (4.5) into the optimal control problem

$$\text{(4.7)} \qquad \begin{cases} \text{Minimize } \displaystyle\int_0^1 t^2[y(t)^2 + u(t)^2]^{\frac{1}{2}}\, dt, \\ u \in L^2[0,1], \\ \displaystyle\int_0^1 u(t)dt = 1. \end{cases}$$

For notational convenience, we introduce the function $\mathbf{1} \in L^2[0,1]$ given by

$$\mathbf{1}(t) = 1 \quad \text{for a.e. } t \in [0,1]$$

and the closed subspace

$$H := \left\{ h \in L^2[0,1] : \int_0^1 h(t)dt = 0 \right\}.$$

We also write

$$\begin{aligned} I(u,y) &= \int_0^1 t^2[y(t)^2 + u(t)^2]^{\frac{1}{2}}\, dt, \\ \Psi_H(h) &= 0 \quad \text{if } h \in H, \quad +\infty \text{ if not}, \\ f(u,y) &= I(u,y) + \Psi_H(u - \mathbf{1}). \end{aligned}$$

With this notation it is clear that m is the marginal function

$$y \in L^2[0,1] \mapsto m(y) = \text{Inf}\{f(u,y) : u \in L^2[0,1]\}.$$

According to Corollary 5, the exact subdifferential of m at $\bar{y}$ is given by

$$\text{(4.8)} \qquad \partial m(\bar{y}) = \bigcap_{\eta>0} \bigcup_{u \in L^2[0,1]} \{v \in L^2[0,1] : (0,v) \in \partial_\eta f(u,\bar{y})\}.$$

Taking into account the specific structure of f, one observes that

$$\partial_\eta f(u,\bar{y}) = \begin{cases} \partial_\eta I(u,\bar{y}) + H^\perp \times \{0\} & \text{if } u \in \mathbf{1}+H, \\ \text{empty set} & \text{if } u \notin \mathbf{1}+H, \end{cases} \tag{4.9}$$

where

$$H^\perp = \left\{ p \in L^2[0,1] : \int_0^1 p(t)h(t)dt = 0 \text{ for all } h \in H \right\}$$

corresponds to the orthogonal space of H. Plugging (4.9) into (4.8) and rearranging terms, one gets

$$\partial m(\bar{y}) = \bigcap_{\eta>0} \bigsqcup_{\substack{u\in 1+H \\ p\in H^\perp}} \{v \in L^2[0,1] : (p,v) \in \partial_\eta I(u,\bar{y})\}.$$

It now remains to evaluate the approximate subdifferential of I. Observe that I is an integral functional of the form

$$I(u,y) = \int_0^1 L_t(u(t),y(t))dt,$$

where

$$(t,\alpha,\beta) \in \mathbb{R} \times (\mathbb{R}\times\mathbb{R}) \mapsto L_t(\alpha,\beta) = t^2[\alpha^2+\beta^2]^{\frac{1}{2}}$$

is a Caratheodory convex integrand (in particular, it is a normal convex integrand in the sense of Rockafellar [23], [24], [25]). According to Bustos [5, Prop. 2.2.1] (see also Hiriart-Urruty [10, §4]), one has

$$(p,v) \in \partial_\eta I(u,\bar{y}) \Leftrightarrow \begin{cases} \text{there exists } \lambda \in \Lambda(\eta) & \text{such that} \\ (p(t),v(t)) \in \partial_{\lambda(t)} L_t(u(t),\bar{y}(t)) & \text{for a.e. } t \in [0,1]. \end{cases}$$

Here we use the notation

$$\Lambda(\eta) := \left\{ \lambda \in L^1[0,1] : \int_0^1 \lambda(t)dt = \eta,\ \lambda(t) \geq 0 \text{ for a.e. } t \in [0,1] \right\}.$$

As a matter of direct computation one also has

$$\partial_{\lambda(t)} L_t(\alpha,\beta) = \{(\gamma,\delta) \in \mathbb{R}^2 : [\gamma^2+\delta^2]^{\frac{1}{2}} \leq t^2,\ \alpha\gamma+\beta\delta \geq t^2[\alpha^2+\beta^2]^{\frac{1}{2}} - \lambda(t)\}.$$

Combining these results and taking into account the particular choice of $\bar{y}$, one finally obtains

$$v \in \partial m(\bar{y}) \Leftrightarrow \begin{cases} \text{for all } \eta > 0 \quad \text{there exists } u \in \mathbf{1}+H, \quad p \in H^\perp, \quad \text{and} \\ \lambda \in \Lambda(\eta) \quad \text{such that, for a.e. } t \in [0,1], \quad \text{one has} \\ [p(t)^2+v(t)^2]^{\frac{1}{2}} \leq t^2 \quad \text{and} \quad p(t)u(t) \geq t^2|u(t)| - \lambda(t). \end{cases}$$

A further simplification is obtained by inspecting more closely the set $H^\perp$. It is clear that $H^\perp$ is the linear subspace of $L^2[0,1]$ spanned by the vector $\mathbf{1}$, i.e.,

$$H^\perp = \{c\,\mathbf{1} : c \in \mathbb{R}\}.$$

Hence,

$$v \in \partial m(\bar{y}) \Leftrightarrow \begin{cases} \text{for all } \eta > 0 \text{ there exists } u \in L^2[0,1],\ c \in \mathbb{R}, \text{ and } \lambda \in \Lambda(\eta) \text{ such that} \\ \text{(i)} \quad \displaystyle\int_0^1 u(t)dt = 1, \\ \text{(ii)} \quad [c^2 + v(t)^2]^{\frac{1}{2}} \leq t^2 \quad \text{for a.e. } t \in [0,1], \\ \text{(iii)} \quad cu(t) \geq t^2|u(t)| - \lambda(t) \quad \text{for a.e. } t \in [0,1]. \end{cases}$$

In particular, a necessary condition for v to be in $\partial m(\bar{y})$ is that

$$|v(t)| \leq t^2 \quad \text{for a.e. } t \in [0,1].$$

Conversely, any function v, as above, belongs to the set $\partial m(\bar{y})$. Indeed, for a given $\eta > 0$, just take $c = 0$ and choose u as an optimal solution within a tolerance η to the unperturbed optimal control problem, and

$$\lambda(t) = \eta t^2|u(t)| \left[\int_0^1 \tau^2|u(\tau)|d\tau\right]^{-1} \quad \text{for a.e. } t \in [0,1].$$

Thus,

$$\partial m(\bar{y}) = \{v \in L^2[0,1] : |v(t)| \leq t^2 \text{ for a.e. } t \in [0,1]\}. \tag{4.10}$$

Finally, observe that a direct evaluation of the optimal value $m(y)$ associated with (4.5) yields

$$m(y) = \int_0^1 t^2|y(t)|dt.$$

This, of course, confirms our estimate (4.10).

4.2. Minimal hypersurfaces and related problems. As in Example 1 consider an open bounded regular set $\Omega \subset \mathbb{R}^n$, and variational problem of the type

$$\begin{cases} \text{Minimize} \displaystyle\int_\Omega L_\omega(\varphi(\omega), \operatorname{grad}\varphi(\omega))d\omega, \\ \varphi \in W^{1,1}(\Omega), \\ \varphi - y \in W_0^{1,1}(\Omega). \end{cases} \tag{4.11}$$

If there is a doubt concerning the notation, the reader is referred to [7, §5], by Ekeland and Temam. Our concern here is to study the sensitivity of the optimal value function

$$h(y) := \inf_{\varphi \in W^{1,1}(\Omega)} \left\{\int_\Omega L_\omega(\varphi(\omega), \operatorname{grad}\varphi(\omega))d\omega : \varphi - y \in W_0^{1,1}(\Omega)\right\}. \tag{4.12}$$

The parameter y is supposed to be an element of $W^{1,1}(\Omega)$. This space will be paired with its topological dual $[W^{1,1}(\Omega)]^*$ by means of the canonical pairing

$$\langle T, y\rangle := T(y) \quad \text{for all } T \in [W^{1,1}(\Omega)]^*, \quad y \in W^{1,1}(\Omega).$$

To proceed further we need to impose some suitable measurability and convexity assumptions on the integrand L_ω. The next set of hypotheses can be substantially weakened, but we are not interested here in discussing these details. We simply posit

(4.13)

$$\begin{cases} \text{(a)} & \text{for every } (\alpha,\xi)\in\mathbb{R}\times\mathbb{R}^n, \quad \omega\in\Omega\mapsto L_\omega(\alpha,\xi) \quad \text{is measurable;} \\ \text{(b)} & \text{for a.e. } \omega\in\Omega, \quad (\alpha,\xi)\in\mathbb{R}\times\mathbb{R}^n\mapsto L_\omega(\alpha,\xi) \quad \text{is convex;} \\ \text{(c)} & \text{there exists } a\in L^1(\Omega;\mathbb{R}_+), \quad b\in\mathbb{R}_+, \quad c\in\mathbb{R}_+ \quad \text{such that} \\ & |L_\omega(\alpha,\xi)|\le a(\omega)+b|\alpha|+c\|\xi\| \quad \text{for all } (\alpha,\xi)\in\mathbb{R}\times\mathbb{R}^n \quad \text{and for a.e. } \omega\in\Omega. \end{cases}$$

In the next proposition we fully characterize the subgradients of h at $\bar{y}$, that is, the elements in the set

$$\partial h(\bar{y})=\{T\in[W^{1,1}(\Omega)]^*: h(y)\ge h(\bar{y})+\langle T, y-\bar{y}\rangle \text{ for all } y\in W^{1,1}(\Omega)\}.$$

We use the notation

$$[W_0^{1,1}(\Omega)]^\perp := \{T\in[W^{1,1}(\Omega)]^*: \langle T,\varphi\rangle = 0 \text{ for all } \varphi\in W_0^{1,1}(\Omega)\}$$

for the orthogonal complement of $W_0^{1,1}(\Omega)$, and we set

$$\sum(\eta)=\left\{\sigma\in L^1(\Omega): \int_\Omega \sigma(\omega)d\omega=\eta,\ \sigma(\omega)\ge 0 \text{ for a.e. } \omega\in\Omega\right\}.$$

PROPOSITION 2. *Suppose the general assumption* (4.13) *holds true, and let the function h be finite at the reference level $\bar{y}\in W^{1,1}(\Omega)$. Then, T is a subgradient of h at $\bar{y}$ if and only if $T\in[W_0^{1,1}(\Omega)]^\perp$, and for all $\eta>0$ there are functions $\varphi\in W^{1,1}(\Omega)$, $p\in L^\infty(\Omega)$, $q\in L^\infty(\Omega)^n$, and $\sigma\in\sum(\eta)$ such that*

$$\varphi-\bar{y}\in W_0^{1,1}(\Omega), \tag{4.14}$$

$$T=p-\operatorname{div} q, \tag{4.15}$$

$$(p(\omega),\, q(\omega))\in\partial_{\sigma(\omega)}L_\omega(\varphi(\omega),\,\operatorname{grad}\varphi(\omega)) \quad \text{for a.e. } \omega\in\Omega. \tag{4.16}$$

Proof. The function h can be written in the form

$$h(y)=f_{A,K}(y)=\operatorname{Inf}\{f(\varphi): y-A\varphi\in K\}.$$

Just take $K=W_0^{1,1}(\Omega)$, A equal to the identity on $W^{1,1}(\Omega)$, and $f: W^{1,1}(\Omega)\to\mathbb{R}\cup\{+\infty\}$ given by $f(\varphi)=(I_L\circ\Lambda)(\varphi)$. Here

$$I_L(\varphi,\psi):=\int_\Omega L_\omega(\varphi(\omega),\psi(\omega))d\omega \quad \text{for all } (\varphi,\psi)\in L^1(\Omega)\times L^1(\Omega)^n,$$

and Λ is the standard operator [7, p. 133]

$$\Lambda: W^{1,1}(\Omega)\to L^1(\omega)\times L^1(\Omega)^n,$$
$$\varphi\mapsto(\varphi,\,\operatorname{grad}\varphi).$$

It suffices, then, to apply Corollary 3 and to keep in mind the following facts: first, $K=W_0^{1,1}(\Omega)$ is a linear subspace and thus $K^+=K^-=[W_0^{1,1}(\Omega)]^\perp$; second, $A^*=A$ and $\Lambda^*: L^\infty(\Omega)\times L^\infty(\Omega)^n\to[W^{1,1}(\Omega)]^*$ is given by $\Lambda^*(p,q)=p-\operatorname{div} q$; third, one can apply the general formula

$$\partial_\eta(I_L\circ\Lambda)(\varphi)=\Lambda^*[\partial_\eta I_L(\Lambda\varphi)] \quad \text{(cf. [9, p. 58])};$$

and finally,

$$(p,q) \in \partial_\eta I_L(\varphi,\psi) \Leftrightarrow \begin{cases} \exists \sigma \in \sum(\eta) \quad \text{such that} \\ (p(\omega),q(\omega)) \in \partial_{\sigma(\omega)} L_\omega(\varphi(\omega),\psi(\omega)) \quad \text{for a.e. } \omega \in \Omega. \end{cases}$$

Joining all the pieces together, one gets the desired results. □

It is worth mentioning that condition (4.16) appears also in Bustos's characterization [5, Cor. 2.3.1] of the approximate solutions to the variational problem (4.11). Observe that condition (4.16) is a pointwise inclusion: it must be satisfied for almost all $\omega \in \Omega$, but each time, ω is regarded as fixed. From a calculus point of view, checking condition (4.16) amounts to evaluating the approximate subdifferential mapping

$$(\alpha,\xi) \in \mathbb{R} \times \mathbb{R}^n \mapsto \partial_\sigma L_\omega(\alpha,\xi)$$

of the given convex integrand L_ω. Proposition 2 applies in particular to the following situations:

$$L_\omega(\alpha,\xi) = [1 + \|\xi\|^2]^{1/2} \quad \text{(minimal hypersurfaces)},$$
$$L_\omega(\alpha,\xi) = [1 + \|\xi\|^2]^{1/2} - r(\omega)\alpha \quad \text{(cf. Mossolov [18])},$$
$$L_\omega(\alpha,\xi) = [1 + \|\omega\|^2 + \|\xi\|^2]^{1/2} \quad \text{(cf. Bernstein [1], [2])}.$$

For additional examples of interesting convex integrands, see Ekeland and Temam [7, p. 136].

Acknowledgments. The authors are grateful to the referees for their careful reading of the manuscript and their constructive comments.

REFERENCES

[1] S. Bernstein, *Sur la généralisation du problème de Dirichlet*, Math. Am., (1910), pp. 82–136.
[2] S. Bernstein, *Sur les équations de calcul des variations*. Ann. Sc. Ecole Norm. Sup., 29 (1912), pp. 431–485.
[3] J. Borwein, *A note on ε-subgradients and maximal monotonicity*, Pacific J. Math., 103 (1982), pp. 307–314.
[4] A. Brondsted and R. T. Rockafellar, *On the subdifferentiability of convex functions*, Proc. AMS, 16 (1965), pp. 605–611.
[5] M. Bustos, *Conditions d'optimalité à ε-près dans un problème d'optimisation non différentiable*, Ph.D. thesis, Université Paul Sabatier, Toulouse, France, 1989.
[6] V. F. Dem'yanov and L. V. Vasil'ev, *Nondifferentiable optimization*, Translation Series in Mathematics and Engineering, Optimization Software, New York, 1985.
[7] I. Ekeland and R. Temam, *Analyse convexe et problèmes variationnels*, Dunod, Paris, 1974.
[8] N. Heukemes, V. H. Nguyen, and J.-J. Strodiot, *ε-optimal solutions in nondifferentiable programming and some related questions*, Math. Programming, 25 (1983), pp. 307–328.
[9] J.-B. Hiriart-Urruty, *ε-subdifferential calculus*, in Research Notes in Mathematics, Vol. 57, Pitman, New York, 1982, pp. 43–92.
[10] ———, *Contributions à la programmation mathématique: Cas déterministe et stochastique*, Ph.D. thesis, Université de Clermont-Ferrand II, France, 1977.
[11] ———, *From convex optimization to nonconvex optimization*, in Nonsmooth Optimization and Related Topics, F. H. Clarke, V. F. Demyanov, and F. Giannessi, eds., Plenum Press, 1989, pp. 219–239.
[12] J.-B. Hiriart-Urruty and A. Seeger, *The second-order subdifferential and the Dupin indicatrices of a nondifferentiable convex function*, Proc. London Math. Soc., 58 (1989), pp. 351–365.
[13] A. D. Ioffe and V. H. Tihomirov, *Theory of extremal problems*, Studies in Mathematics and Its Applications, Vol. 6, North-Holland, Amsterdam, 1979.
[14] C. Lemarechal and J. Zowe, *Some remarks on the construction of higher-order algorithms in convex optimization*, Appl. Math. Optim., 10 (1983), pp. 51–68.
[15] L. McLinden, *An application of Ekeland's theorem to minimax problems*, Nonlinear Anal., 6 (1982), pp. 189–196.
[16] ———, *A minimax theorem*, Math. Oper. Res., 9 (1984), pp. 516–591.
[17] ———, *Quasistable parametric optimization without compact level sets*, Technical Summary Report #2708, Mathematics Research Center, Univ. of Wisconsin-Madison, June 1984.

[18] P. P. Mossolov, *Sur une fonctionnelle liée aux surfaces de courbure moyenne fixée*, Mat. Sbornik U.S.S.R., 78 (1969), pp. 51–64.
[19] W. Oettli, *Epsilon-solutions and epsilon-supports*, Optimization, 16 (1985), pp. 491–496.
[20] S. M. Robinson, *Bundle-based decomposition: Conditions for convergence*, in Analyse Nonlinéaire, H. Attouch, J.-P. Aubin, F. Clarke, and I. Ekeland, eds., Gauthier-Villars, Paris, 1989.
[21] R. T. Rockafellar, *Conjugate Duality and Optimization*, SIAM Regional Conference Series in Applied Mathematics, Vol. 16, Society for Industrial and Applied Mathematics, Philadelphia, 1974.
[22] ———, *Convex Analysis*, Princeton University Press, Princeton, NJ, 1970.
[23] ———, *Integrals which are convex functionals*, Pacific J. Math., 24 (1968), pp. 525–540.
[24] ———, *Measurable dependence of convex sets and functions on parameters*, J. Math. Anal. Appl., 28 (1969), pp. 4–25.
[25] ———, *Convex integral functionals and duality*, in Contributions to Nonlinear Functional Analysis, E. Zaratonello, ed., Academic Press, New York, 1971, pp. 215–236.
[26] A. Seeger, *Analyse du second ordre de problèmes non différentiables*, Ph.D. thesis, Université Paul Sabatier, Toulouse, France, 1986.
[27] ———, *Complément de Schur et sous-différentiel du second-ordre d'une fonction convexe*, Aequationes Math., 42 (1991), pp. 47–71.
[28] ———, *Limiting behaviour of the approximate second-order subdifferential of a convex function*, J. Optim. Theory Appl., 74 (1992), pp. 527–544.
[29] ———, *Approximate Euler-Lagrange inclusion, approximate transversality condition, and sensitivity analysis of convex parametric problems of calculus of variations*, Preprint, Department of Mathematics, University of Avignon, France, May 1992.
[30] S. Shiraishi, *First-order and second-order ε-directional derivatives of a marginal function in convex programming with linear inequality constraints*, J. Optim., Theory Appl., 66 (1990), pp. 489–502.
[31] C. Zalinescu, *Stability for a class of nonlinear optimization problems and applications*, in Nonsmooth Optimization and Related Topics, F. H. Clarke, V. F. Demyanov, F. Giannessi, eds., Plenum Press, New York, 1989, pp. 437–458.
[32] J. Zowe, *Nondifferentiable optimization— A motivation and a short introduction into the subgradient and the bundle concept*, in Computational Mathematical Programming, K. Schittkowski, ed., Springer-Verlag, Berlin, 1985.

SIAM J. OPTIMIZATION
Vol. 4, No. 3, pp. 676–689, August 1994
© 1994 Society for Industrial and Applied Mathematics
015

OUTER-FACIAL GRAPHS AND THE TRAVELING SALESMAN PROBLEM*

DAVID HARTVIGSEN† AND W. R. PULLEYBLANK‡

Abstract. An outer-facial graph is a planar 3-connected graph with one face having a node or edge in common with every other face. It is shown that the traveling salesman problem (TSP) in such graphs, for arbitrary edge weights, can be transformed into a variant of the TSP (called the international salesman problem (ISP)) in an associated Halin graph. In its simplest form, an ISP is obtained by partitioning the nodes of a graph into a set of *countries*. A tour of minimum cost that visits each country exactly once is required, but within each country a visit may be made to any nonempty subset of the nodes.

A linear system is described sufficient to define the convex hull of the incidence vectors of the feasible tours for an ISP obtained in this fashion. This linear system has size polynomial in the size of the original graph. Thus, the TSP in an outer-facial graph is reduced to a small linear programming problem.

The traveling salesman polytope of an outer-facial graph can be obtained as a projection of the polyhedron defined by the above system. For a class of outer-facial graphs, this fact is used to give explicitly a defining system for the TSP. This system includes the well-known subtour elimination constraints and comb inequalities.

Key words. traveling salesman problem, polyhedral combinations, planar graphs

AMS subject classifications. 05, 90

1. Introduction.

Suppose we are given a graph $G = (V, E)$ and a vector $c = (c_j : j \in E)$ of real edge costs. The TSP is to find a hamiltonian cycle in G for which the sum of the edge costs is minimized. (A hamiltonian cycle passes exactly once through each node of G.) Cornuejols, Naddef, and Pulleyblank [3] discussed this problem for several classes of graphs and showed that for these classes, the problem can be solved in the following sense:

(a) a minimum cost hamiltonian cycle can be obtained in time polynomial in the size of G;

(b) a linear system whose solution set is the convex hull of the set of incidence vectors of the hamiltonian cycles can be explicitly described.

The methods in [3] were based upon a reduction that could be performed when the graph contained 3-edge cutsets. By shrinking each of the shores of a 3-edge cutset in turn, the problem was reduced to several smaller problems such that solutions to (a) and (b) for these smaller problems could be combined to yield solutions to (a) and (b) for the original problem. Thus the original problem could be solved provided that (a) and (b) could be solved for the irreducible, i.e., 3-edge cutset free, graphs obtained in this process. For several classes of graphs, the irreducible graphs had a simple structure and (a) and (b) could be solved for all graphs in the class.

One class of graphs for which the TSP was solved in [3] is the class of Halin graphs. A Halin graph, $H = T \cup C$ is formed by taking a tree T having no degree two nodes and adding edges to form a cycle C that joins the degree-one nodes (or leaves) in such a way that the resulting graph is planar. See Fig. 1.1.

* Received by the editors April 16, 1993; accepted for publication (in revised form) November 17, 1993.

† Department of Management, College of Business Administration, University of Notre Dame, Notre Dame, Indiana 46556. This research was supported in part by the Natural Sciences and Engineering Research Council of Canada.

‡ Department of Combinatorics and Optimization, University of Waterloo, Waterloo, Ontario, Canada; Author's current address: Mathematical Sciences Department, T. J. Watson Research Center, IBM Corporation, Yorktown Heights, NY 10598. This research was supported in part by the Natural Sciences and Engineering Research Council of Canada.

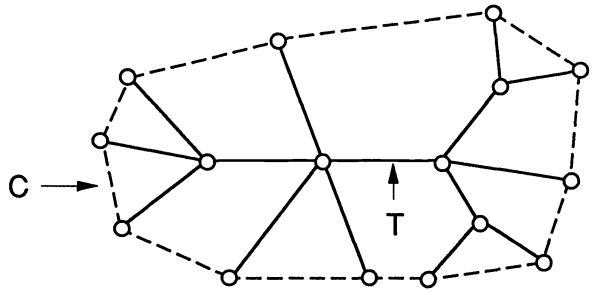


FIG. 1.1. *A Halin graph.*

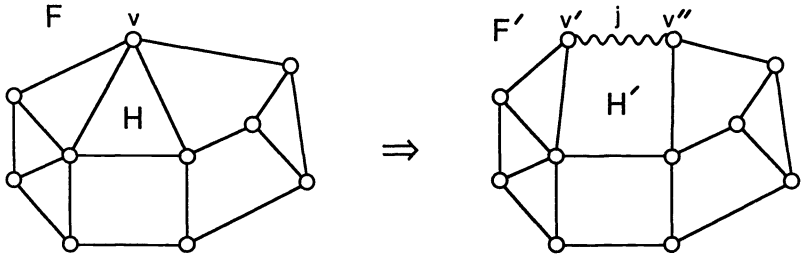


FIG. 1.2

It is easy to see that the Halin graphs are 3-connected and that their planar embeddings are unique up to the choice of the outer face. In fact, they are precisely those 3-connected simple planar graphs that can be embedded in such a way that every face has an *edge* in common with the outer face. The irreducible Halin graphs are the *wheels*, obtained when T contains a single nonleaf node.

We say that a simple 3-connected planar graph G is *outer-facial* if it can be embedded in such a way that each face has at least one *node* in common with the outer face. We make use of the following (well-known) correspondence between outer-facial graphs and Halin graphs: let $H = T \cup C$ be a Halin graph and let B be a subset of the edges of C, none of which belongs to a triangular face. If we contract the members of B, the resulting graph G is then outer-facial.

Conversely, let G be an outer-facial graph, let F be the outer face, and let H be any face not possessing an edge in common with F. Let v be the node that H and F have in common. Split v into nodes v' and v'' joined by a new edge j, such that j will be an edge of the faces H' and F' of the resulting graph, containing, respectively, the edges of H and F. (See Fig. 1.2.)

If we perform this operation for every such face H, then we obtain a Halin graph we call the *associated Halin graph of* G, and denote the graph by $H(G)$.

The nodes of $H(G)$ are of two types: some are nodes of G; we call these *real* nodes. Others were created by splitting; we call these *split* nodes. Each such node is associated with a unique node of G. For any such node v of G, we let $B(v)$ denote the set of split nodes of $H(G)$ associated with v. Note that the subgraph of $H(G)$ induced by $B(v)$ will be a path, all of whose edges were introduced by the splitting operation. We let B denote the set of edges introduced by the splitting operation.

Let C be a hamiltonian cycle in G. The edges of C can be extended in a unique way to a cycle C' in $H(G)$ by adding edges of B. This cycle will not generally be hamiltonian. Let $\mathcal{P}$ denote the set of cycles C' of $H(G)$ such that the edges of C' that belong to G constitute the

edge set of a hamiltonian cycle of G. If we define $c_j = 0$ for all $j \in B$, then the TSP for G is equivalent to the problem of finding $C \in \mathcal{P}$ for which $\sum(c_j : j \in E(C))$ is minimized.

PROPOSITION 1.1. *Let G be an outer-facial graph and let $H = H(G)$. $C \in \mathcal{P}$ if and only if*

(i) *every real node of H belongs to C;*

(ii) *for each set $B(v)$ of split nodes of H, the intersection of C with $H[B(v)]$ is a simple (no repeated nodes) nonempty path.*

Proof. The proof of the proposition is evident. □

In §2 we define a useful reduction that can be applied to Halin graphs $H(G)$. We then examine what hamiltonian cycles in G correspond to in $H(G)$ after it has been reduced. In §3 we describe a linear system sufficient to define the convex hull of the incidence vectors of the members of $\mathcal{P}$ for $H(G)$. In other words, we reduce the TSP in the original outer-facial graph to a linear programming problem, whose variables correspond to the edges of $H(G)$. The number of constraints grows polynomially with the size of G.

In §4 we discuss how Gaussian and Fourier–Motzkin elimination can be used to eliminate the variables corresponding to edges in B, and thereby obtain the traveling salesman polytope for an outer-facial graph. Unfortunately, these resulting systems seem to be extremely complex. However, for "simple" outer-facial graphs we explicitly describe a linear system sufficient to define the traveling salesman polytope. An outer-facial graph is *simple* if, whenever each of faces F_1 and F_2 shares exactly one node with the outer face, and these nodes are distinct, then F_1 and F_2 have no edge in common.

In the final section we discuss a generalization of the TSP, called the ISP. The problem we discussed for $H(G)$ is an instance of this. In the Appendix we briefly describe how to obtain a linear time algorithm that solves the TSP on outer-facial graphs.

A main goal of our research is to find a complete linear programming formulation for a special class of a hard combinatorial optimization problem. This is of theoretical interest in the field of polyhedral combinatorics, which studies the structure of polyhedra of combinatorial optimization problems. In addition, this work has practical implications. This is due to the success with which many hard combinatorial optimization problems have recently been attacked using partial linear programming (LP) characterizations together with cutting plane techniques (see, e.g., the work of Padberg and Rinaldi on the TSP [12]).

A complete LP characterization for a special case of a hard problem tells us that the inequalities in this characterization are generally important: they will probably be useful in any attempt to solve this problem on more general examples using cutting plane techniques. For example, in this paper our complete LP characterization for a class of outer-facial graphs contains subtour elimination and comb inequalities, which were indeed useful in the work done in [12]. In addition, if we want to add more inequalities to our partial LP characterization, in an effort to strengthen our cutting plane approach, then such inequalities will often be derived from structures at least as complex as those in the special case. Hence these results may indicate where to look for such inequalities. Finally, looking at a special case may lead to the discovery of new classes of inequalities. For example, due to the projection technique used in this paper, we have found an apparently new class of inequalities for the TSP, the so-called "island constraints."

2. Outer-facial graphs. We begin this section by making some useful definitions. We then examine the structure of hamiltonian cycles in outer-facial and some related classes of graphs.

We let $G = (V, E)$ denote a graph with node set V and edge set E. If an edge $e \in E$ is incident with nodes $u, v, \in V$, then we write $e = uv$. For any $S \subseteq V$, we let $\delta(S)$ denote the set of edges with one end-node in S and the other in $V - S$; that is, $\delta(S) \equiv \{uv \in E : u \in S,$

$v \notin S\}$. For every $v \in V$, we abbreviate $\delta\{(v)\}$ by $\delta(v)$. When S and $V - S$ each contain at least two nodes, we call $\delta(S)$ a *nontrivial edge cutset*, or simply an *edge cutset*, and call S and $V - S$ its *shores*. If $|\delta(S)| = 3$, then we call $\delta(S)$ a 3-*edge cut*. (For $v \in V$, we call $\delta(v)$ a *trivial edge cutset*.) For $S \subseteq V$, we let $G \times S$ denote the graph obtained by *shrinking* (or contracting) S. That is, the nodes of $G \times S$ are all nodes of $V - S$, plus a new pseudonode $\tilde{S}$ obtained by identifying all nodes of S. The edges of $G \times S$ are defined as follows:

(i) An edge with both ends in S disappears.

(ii) An edge with both ends in $V - S$ remains unchanged.

(iii) An edge of $\delta(S)$ now joins the incident node of $V - S$ and the pseudonode $\tilde{S}$.

For any $S \subseteq V$, we let $\gamma(S)$ denote the set of edges of G having both ends in S and we let $G[S]$ denote the subgraph of G induced by the node set S; that is, $G[S] \equiv (S, \gamma(S))$. If $x = (x(e) : e \in E)$ is any function from E into $\mathbb{R}$, then for any $\tilde{E} \subseteq E$ we define $x(\tilde{E}) \equiv \sum(x(e) : e \in \tilde{E})$.

Let $H = T \cup C$ be a Halin graph. If T is a star, then H is a *wheel*. If T is not a star, then there exists a nonleaf node v of T adjacent only to leaves of T plus one other nonleaf node. Let S consist of v plus all adjacent leaves of T. We call $H[S]$ a *fan*. Note that $|\delta(S)| = 3$ and $H \times S$ is a Halin graph. In fact, shrinking a shore associated with any 3-edge cut in a Halin graph results in a new Halin graph. It follows that a Halin graph can always be reduced to a wheel by shrinking a sequence of shores defined by nontrivial 3-edge cuts. In the Appendix we briefly describe how this idea can be used to construct a linear time algorithm for solving the traveling salesman problem in outer-facial graphs.

Suppose we start with an outer-facial graph $G = (V, E)$. Let $H = H(G)$ be the associated Halin graph. After we have performed some shrinking operations defined by 3-edge cutsets on H, our nodes are of three types:

(i) *real nodes*, which are nodes of G;

(ii) *split nodes*, which are nodes of H but not of G; and

(iii) *pseudonodes*, which were created from sets of nodes of H by shrinking operations.

Let $\tilde{H}$ be such a Halin graph obtained from H by a sequence of shrinkings defined by 3-edge cutsets. A *city-state* in $\tilde{H}$ is a real node or pseudonode. Let v be a node of G which was split, forming $B(v)$, in H. Let $I(v)$ be the set of split nodes of $\tilde{H}$ belonging to $B(v)$, plus any pseudonodes of $\tilde{H}$ that contain nodes of $B(v)$. If $|I(v)| \geq 2$, then $I(v)$ is called a *country*. (If $|I(v)| = 1$, then $I(v)$ contains a single pseudonode, and hence is a city-state.)

Countries contain only split nodes and pseudonodes. A split node can only occur in a single country. A pseudonode may occur in zero, one, or two countries. Moreover, no country can properly contain another, although two may intersect in a pseudonode city-state and counties may contain pseudonodes. (In Fig. 2.1, $\tilde{H}$ was obtained from $H(G)$ by shrinking the fan with node set $\{a, b_1, e_1, f, g\}$ in $H(G)$ to $\tilde{v}$. In this example, $\tilde{H}$ has two countries, $\{\tilde{v}, b_2, b_3\}$ and $\{\tilde{v}, e_2\}$. Nodes $\tilde{v}$, c, d, h, and i are the city-states of $\tilde{H}$. Observe that $B(b) = \{b_1, b_2, b_3\}$ in $H(G)$ and $I(b) = \{\tilde{v}, v_2, b_3\}$ in $\tilde{H}$.

Countries in $\tilde{H}$ have the following properties:

(i) They induce edge disjoint paths.

(ii) Interior nodes of these paths are split nodes.

(iii) Endnodes of these paths are either pseudo- or split nodes.

(iv) Any node common to two countries is a pseudonode.

(v) They occur along the outer face of the graph.

If the graph has been shrunk to a wheel, countries have the following additional structures:

(vi) Each country has two or three nodes.

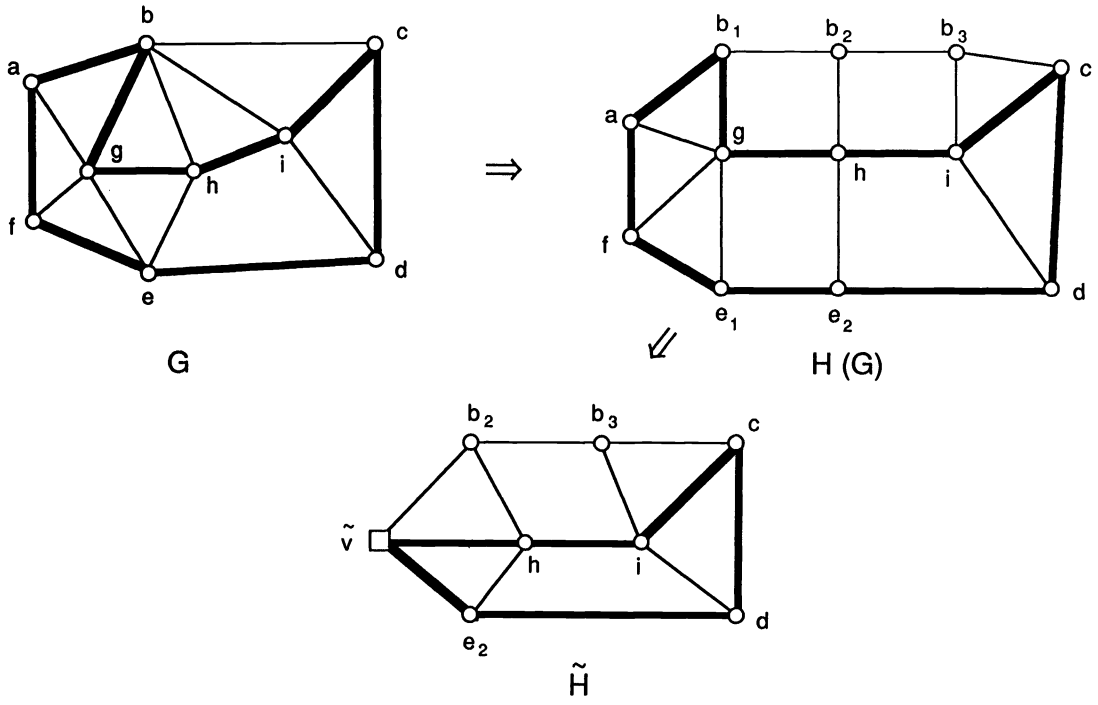


FIG. 2.1

(vii) If a country has two nodes, then at least one must be a pseudonode.

(viii) If a country has three nodes, then both endnodes of the induced path must be pseudonodes.

These properties follow from property (ii) and the observation that no two adjacent nodes in a country on a wheel can be split nodes. Otherwise the two radial edges incident with these nodes would have been parallel in the original graph, which we disallow.

(ix) There can be at most two consecutive split nodes along the rim of a wheel, in which case they must occur in different countries.

For any hamiltonian cycle C of G, we let $\tilde{C}$ be the cycle in $\tilde{H}$ whose edge set consists of the edges of C in $\tilde{H}$. Note that $\tilde{C}$ need not be a hamiltonian cycle in $\tilde{H}$. We let $\tilde{\mathcal{P}}$ be the set of all such cycles $\tilde{C}$ in $\tilde{H}$ (see Fig. 2.1, where the edges of C and $\tilde{C}$ are indicated by bold lines). We are now able to characterize $\tilde{\mathcal{P}}$ in terms of the countries and city-states of $\tilde{H}$.

PROPOSITION 2.1. *Let $\tilde{C}$ be a cycle in $\tilde{H}$. Then $\tilde{C} \in \tilde{\mathcal{P}}$ if and only if*

(i) *every city-state $\tilde{H}$ belongs to $\tilde{C}$;*

(ii) *at least one node of each country belongs to $\tilde{C}$;*

(iii) *if $\tilde{B}$ is the set of split nodes of some country of $\tilde{H}$, then the intersection of $\tilde{C}$ with $\tilde{H}[\tilde{B}]$ is a single path or empty.*

Proof. The proof follows easily by induction on the number of shrinkings performed in obtaining $\tilde{H}$ from H, and from Proposition 1.1. □

Conditions (i)–(iii) of Proposition 2.1 force $\tilde{C}$ to visit each country and city-state exactly once, provided we count a "visit" as follows. First, $\tilde{C}$ contains each city-state exactly once, and we count this as the visit to the city-state. Second, for each country K, the intersection of $\tilde{C}$ with $\tilde{H}[K]$ consists of one or more segments, at most one of which can contain a split node of K. (Note that this, together with properties (ii) and (iii) ensure that at most one segment can contain an edge with both ends in K.) If such a segment exists, it is counted as the visit to the country. If not, any (singleton) segment can be so counted.

We call members of $\tilde{\mathcal{P}}$ the *feasible cycles* of $\tilde{H}$.

3. Polyhedral descriptions. Let $\tilde{H}$ be a Halin graph with city-states and countries (as described in §2). Associate a variable with each edge of $\tilde{H}$ and a 0–1 incidence vector with each feasible cycle. Let $P(\tilde{H})$ denote the convex hull of these incidence vectors. In this section we describe $P(\tilde{H})$ with a system of linear inequalities and equalities. The size of this system is polynomial in the size of $\tilde{H}$. Hence, if $\tilde{H}$ was obtained from an outer-facial graph G, then the size of this system is also polynomial in the size of G. In other words, we can reduce a TSP on an outer-facial graph to a small linear programming problem.

Our strategy is to first describe $P(\tilde{H})$ when $\tilde{H}$ is a wheel, and to then use this as a basis for an inductive proof of the general case. We begin by stating a related result from [4]. Let $P^c(G)$ denote the convex hull of incidence vectors of cycles of a graph G.

THEOREM 3.1. *If G is a wheel, then $P^c(G)$ is given by the following.*

$$x(\delta(v)) \leq 2, \text{ for all } v \in V. \tag{1}$$

$$x(\delta(v)\backslash\{j\}) - x_j \geq 0, \text{ for all } v \in V \text{ and } j \in \delta(v). \tag{2}$$

$$x(\delta(c)) + x(\delta(v)\backslash\{j\}) - x_j \geq 2, \text{ for all } v \in V\backslash\{c\}, \tag{3}$$

where c is the center node of G and j is the spoke incident to v.

We next use this to describe $P(\tilde{H})$ for wheels.

THEOREM 3.2. *If $\tilde{H} = (V, E)$ is a wheel, then $P(\tilde{H})$ is given by the following.*

$$x(\delta(v)) = 2, \text{ for all city-states } v \in V. \tag{4}$$

$$x(\delta(v)) \leq 2, \text{ for all noncity-states } v \in V. \tag{5}$$

$$x(\delta(v)\backslash\{j\}) - x_j \geq 0, \text{ for all } v \in V \text{ and } j \in \delta(v). \tag{6}$$

Proof. Clearly, the system defined by (3), (4), (5), and (6) contains $P(\tilde{H})$. In fact, every cycle of $\tilde{H}$ whose incidence vector satisfies this system is feasible. This system can be obtained from the system defined by (1), (2), and (3) by setting certain inequalities in (1) to equalities. This implies that the polyhedron defined by (3), (4), (5), and (6) is a face of $P^c(\tilde{H})$. So the extreme points of this polyhedron are feasible cycles of $\tilde{H}$; i.e., the system (3), (4), (5), and (6) describes $P(\tilde{H})$.

Let c denote the center of G and note that c is a city-state. Then each constraint (3) is the sum of the constraints $x(\delta(c)) = 2$ and $x(\delta(v)\backslash\{j\}) - x_j \geq 0$ where $j \in \delta(c)$. Hence, the inequalities of type (3) are redundant. □

THEOREM 3.3. *If $\tilde{H} = (V, E)$ is a Halin graph, then $P(\tilde{H})$ is given by the following.*

$$x(\delta(v)) = 2, \text{ for all city-states } v \in V. \tag{7}$$

$$x(S) = 2, \text{ for all 3-edge cuts } S. \tag{8}$$

$$x(\delta(v))\backslash\{j\}) - x_j \geq 0, \text{ for all } v \in V \text{ and } j \in \delta(v). \tag{9}$$

$$x(S\backslash\{j\}) - x_j \geq 0, \text{ for all 3-edge cuts } S \text{ and } j \in S. \tag{10}$$

$$x(\delta(K)) = 2, \text{ for all countries } K \text{ with all split nodes.} \tag{11}$$

(12) $x(\delta(K')) \leq 2$, *for all countries* K *where* K' *consists of split nodes of* K *that induce a path.*

Proof. The theorem is proved by induction on $|V|$. The smallest $|V|$ can be is 4, in which case $\tilde{H}$ is a wheel and the result follows from Theorem 3.2. Suppose $|V| > 4$ and that the result holds for graphs with fewer nodes. Again, if $\tilde{H}$ is a wheel, we are done, so let us assume $\tilde{H}$ is not a wheel.

Clearly the system (7)–(12) contains $P(\tilde{H})$. We show that any point $\hat{x}$ that satisfies (7)–(12) can be expressed as a convex combination of incidence vectors of feasible cycles of $\tilde{H}$.

Since $\tilde{H}$ is not a wheel, it contains a nontrivial 3-edge cut $S = \{i, j, k\}$. Hence, $\hat{x}(S) = 2$. Let $\tilde{H}^1$ and $\tilde{H}^2$ be obtained from $\tilde{H}$ by shrinking each shore of S. Let $\hat{x}^1$ and $\hat{x}^2$ be the restrictions of $\hat{x}$ to $\tilde{H}^1$ and $\tilde{H}^2$, respectively. Clearly, $\hat{x}^1$ and $\hat{x}^2$ satisfy (7)–(12) for $\tilde{H}^1$ and $\tilde{H}^2$. Hence, by inductive hypothesis, $\hat{x}^1$ and $\hat{x}^2$ can be expressed as convex combinations of feasible cycles in $\tilde{H}^1$ and $\tilde{H}^2$, respectively. Let $\mathcal{P}^1$ and $\mathcal{P}^2$ denote these cycles in $\tilde{H}^1$ and $\tilde{H}^2$, respectively.

Let $\mathcal{P}_i^1$ be the cycles in $\mathcal{P}^1$ that do not contain edge i. Let α_i^1 be the sum of the coefficients on the cycles in $\mathcal{P}_i^1$ in the convex combination. Define $\mathcal{P}_j^1$, $\mathcal{P}_k^1$, $\mathcal{P}_i^2$, $\mathcal{P}_j^2$, $\mathcal{P}_k^2$, α_j^1, α_k^1, α_i^2, α_j^2, and α_k^2, similarly. Note that because $\hat{x}^1$ and $\hat{x}^2$ agree on x_i, x_j and x_k, $\alpha_i^1 = \alpha_i^2$, $\alpha_j^1 = \alpha_j^2$, and $\alpha_k^1 = \alpha_k^2$, so let us simply refer to α_i, α_j and α_k. We have that $\alpha_i + \alpha_j + \alpha_k = 1$.

Our objective is to combine cycles in $\mathcal{P}^1$ and $\mathcal{P}^2$ to yield feasible cycles in $\tilde{H}$ such that $\hat{x}$ can be expressed as a convex combination of these cycles. We do this in three stages, combining the cycles in $\mathcal{P}_i^1$ with $\mathcal{P}_i^2$, $\mathcal{P}_j^1$ with $\mathcal{P}_j^2$, and $\mathcal{P}_k^1$ with $\mathcal{P}_k^2$. Let us consider the case for edge i, the other cases being analogous.

If i is not in a country, then clearly we can combine cycles in $\mathcal{P}_i^1$ and $\mathcal{P}_i^2$ to obtain feasible cycles in $\tilde{H}$ with coefficients that sum to α_i. We next consider the case that i is in a country.

Consider the set N of all split nodes in the country containing i in $\tilde{H}$. We can divide the cycles in $\mathcal{P}_i^1$ into those that contain an edge of $\delta(N)$, say $\bar{\mathcal{P}}_i^1$ and those that do not, say $\tilde{\mathcal{P}}_i^1$. Let $\bar{\alpha}_i^1$ and $\tilde{\alpha}_i^1$ denote the sum of the coefficients on the cycles in $\bar{\mathcal{P}}_i^1$ and $\tilde{\mathcal{P}}_i^1$, respectively. We may analogously define $\bar{\mathcal{P}}_i^2$, $\tilde{\mathcal{P}}_i^2$, $\bar{\alpha}_i^2$ and $\tilde{\alpha}_i^2$. Hence, $\bar{\alpha}_i^1 + \tilde{\alpha}_i^1 = \bar{\alpha}_i^2 + \tilde{\alpha}_i^2 = \alpha_i$.

CLAIM. *The claim is* $\bar{\alpha}_i^1 \leq \tilde{\alpha}_i^2$ *and* $\bar{\alpha}_i^2 \leq \tilde{\alpha}_i^1$.

Proof of claim. Suppose $\bar{\alpha}_i^1 > \tilde{\alpha}_i^2$. $\hat{x}(\delta(N))$ may be expressed in terms of the α's as follows:

$$\begin{aligned}
\hat{x}(\delta(N)) &= 2\bar{\alpha}_i^1 + 2\bar{\alpha}_i^2 + 2\alpha_j + 2\alpha_k \\
&= 2\big(\bar{\alpha}_i^1 + \bar{\alpha}_i^2 + \alpha_j + \alpha_k\big) \\
&> 2(\tilde{\alpha}_i^2 + \bar{\alpha}_i^2 + \alpha_j + \alpha_k) \\
&= 2(\alpha_i + \alpha_j + \alpha_k) = 2,
\end{aligned}$$

which violates the inequality (11) or (12) that corresponds to $\delta(N)$. Hence, the claim follows.

By the definition of feasibility, each cycle in $\bar{\mathcal{P}}_i^1$ can be combined with each cycle in $\tilde{\mathcal{P}}_i^2$ to form a feasible cycle in $\tilde{H}$; by the claim, coefficients can be assigned to these resulting cycles that sum to $\bar{\alpha}_i^1$. Similarly, every cycle in $\bar{\mathcal{P}}_i^2$ can be combined with a cycle in $\tilde{\mathcal{P}}_i^1$ to form feasible cycles in $\tilde{H}$ whose coefficients sum to $\bar{\alpha}_i^2$. The remaining coefficients that sum to α_i can be applied to feasible cycles in $\tilde{H}$ formed from cycles in $\tilde{\mathcal{P}}_i^1$ and $\tilde{\mathcal{P}}_i^2$. Repeating this process for $\mathcal{P}_j^1$ and $\mathcal{P}_j^2$, and $\mathcal{P}_k^1$ and $\mathcal{P}_k^2$ finishes the proof. □

If $\tilde{H} = H(G)$ is the original Halin graph obtained from the outer-facial graph G, there are no pseudonodes, and we obtain the following simplification.

COROLLARY 3.4. *If $\tilde{H} = (V, E) = H(G)$ is the associated Halin graph of an outer-facial graph G, and $\mathcal{P}$ is the set of feasible cycles, then the convex hull of the incidence vectors of the members of $\mathcal{P}$ is given by* (7)–(11) (*where we have a constraint* (11) *for every country*).

Proof. All we need to show is that, in this case, the constraints (12) are implied. Let K' be a subset of a country K such that K' induces a path. Add the constraint (11) $x(\delta(K)) = 2$ to the negatives of the constraints (9) for all $v \in K \backslash K'$, where in each case we take $j \in \delta(v)$ to be the edge of the rim nearest to K'. We obtain $x(\delta(K')) \leq 2$. □

Note that the system of Theorem 3.3 does not include the trivial inequalities $0 \leq x_j \leq 1$ for $j \in E$. These are implied. Let v be an endnode of the edge j. If v is a city-state, then the constraint $x_j \leq 1$ is obtained by subtracting the constraint (9) from the degree constraint (7). If v is a split node then $x_j \leq 1$ is obtained by subtracting the constraint (9) from the constraint (12), where $v = K'$. The constraint $x_j \geq 0$ is obtained by subtracting the constraints $x_k \leq 1$, $x_l \leq 1$ from (8), where $S = \{j, k, l\}$.

4. Projecting to obtain the traveling salesman polytope for outer-facial graphs. Let G be an outer-facial graph and let $P(G)$ be the convex hull of the incidence vectors of hamiltonian cycles of G. In this section we give a linear system that describes $P(G)$ for a simple outer-facial graph G.

Given an outer-facial graph G, let H be the associated Halin graph. Our procedure is as follows. From the results of the previous section, we have a linear description of $P(H)$, the convex hull of the incidence vectors of feasible cycles of H. We then contract, one by one, the new edges of H until we obtain G. With each contraction we successively modify $P(H)$ until we obtain $P(G)$. In other words, we remove or "project away" the variables of $P(H)$ that correspond to the new edges of H.

There are two tools that can be used to project away a set of variables in a linear system of equalities and inequalities. If a variable is contained in an equation of the system, then we can simply solve the equation for this variable and eliminate it from the remaining equalities and inequalities by substitution. This is the technique we use in this section. If a variable does not occur in any equation, then the variable can be eliminated using Fourier–Motzkin elimination (see [2], [5], and [11]). These tools can, in principle, be used to obtain $P(G)$ for any outer-facial graph G. The main difficulty is to obtain a closed form description of the system obtained when the projection has been completed. For the general class of outer-facial graphs, this system appears to be quite complex. In this section we present a closed form description of this projected system for a special class of outer-facial graphs. This class properly contains the Halin graphs for which a linear system description of $P(G)$ was given in [3].

Let us consider three types of nontrivial disconnecting sets in an outer-facial graph G (i.e., set of nodes and/or edges whose removal results in a graph with two components each containing at least two nodes). Let T denote the tree in the associated Halin graph of G. A 3-0-cut is a set of three edges whose removal nontrivially disconnects G. A 2-1-*cut* is a set consisting of two edges, one of which occurs in T, and one node whose removal nontrivially disconnects the graph, but for which the edges are not contained in a 3-0-cut. A 1-2-*cut* is a set of one edge in T and two nodes whose removal nontrivially disconnects the graph, but for which the edge is not contained in a 3-0-cut or a 2-1-cut. See Fig. 4.1.

In this section we deal with the outer-facial graphs that have 3-0-cuts and 2-1-cuts, but no 1-2-cuts. These are the *simple* outer-facial graphs defined in the Introduction. In the course of the projection, several additional types of constraints appear, which we next describe.

Suppose we have a 2-1-cut with node v and edges k and l, where k does not belong to the outer face. See Fig. 4.2. There is a unique face F_k of G that contains both v and k. There may in general be other 2-1-cuts k', l', v such that k' belongs to F_k. In this case we have the

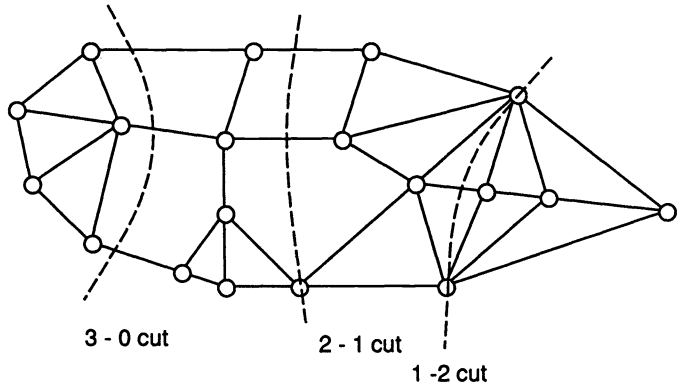


FIG. 4.1

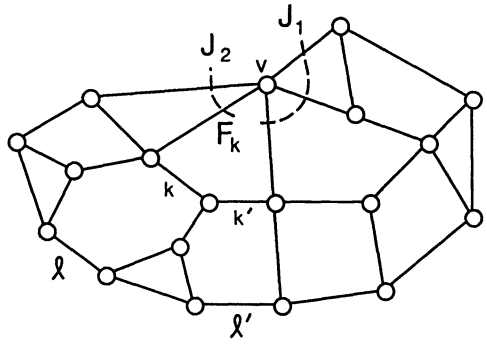


FIG. 4.2

valid equation

$$x_k + x_l - x_{k'} - x_{l'} = 0. \tag{13}$$

The validity of (13) can be seen directly as follows: every hamiltonian cycle C divides the plane into two regions; we call the *inner* region that part that does not contain the outer face. If F_k is in the inner region, then exactly one of $\{k, l\}$ and exactly one of $\{k', l'\}$ belongs to C. If F_k is not in the inner region, then all of $\{k, l, k', l'\}$ belong to C. All such 2-1-cuts that are related by an equation (13) are said to be *equivalent*.

Again, consider the 2-1-cut $\{k, l, v\}$. Removing v, k and l from G, we obtain a graph with two components, K_1 and K_2. Let J_1 be the set of edges joining v to nodes in K_1 and let J_2 be the set of edges joining v to nodes in K_2. See Figure 4.2. Then $J_1 \cup \{k, l\}$ and $J_2 \cup \{k, l\}$ are cuts in G, and we have the valid subtour elimination constraints

$$x(J_i) + x_k + x_l \geq 2 \quad \text{for } i = 1, 2. \tag{14}$$

If $\{k, l, v\}$ and $\{k', l', v\}$ are equivalent 2-1-cuts, then we obtain an equivalent inequality

$$x(J_i) + x_{k'} + x_{l'} \geq 2$$

by subtracting (13) from (14). So it is sufficient to keep only one of these inequalities.

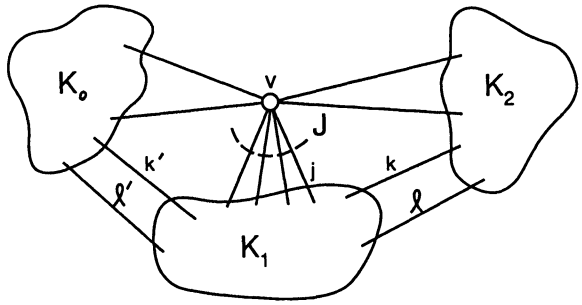


FIG. 4.3

Now choose any $j \in J_i$ and consider the inequality

$$x_j + x_k + x_l - x(J_i \backslash j) \leq 2. \tag{15}$$

This is a form of comb inequality (see Grotschel and Padberg, [6] and [7]). In this case, the ends of the edges j, k, and l form three two-node teeth, and either K_i or $V \backslash K_i$ is the handle. (In fact, these are a special case of 2-matching constraints—a comb constraint for which all teeth have exactly two nodes.) To see that (15) is valid, note that every hamiltonian cycle must contain an even number of edges of the cut $J_i \cup \{k, l\}$. The only way that (15) could be violated would be if all of $\{j, k, l\}$ were used, but in this case a fourth edge, having coefficient -1 in (15), would be used, so (15) would still hold.

We require one additional type of comb inequality. Let $\{k, l, v\}$ be a 2-1-cut and let $\{k', l', v\}$ be a nonequivalent 2-1-cut. (See Fig. 4.3.) If we delete v, k, l, k', and l' we obtain three components K_0, K_1, and K_2, where K_0 contains ends of k', l'; K_2 contains ends of k and l; and K_1 contains ends of all of k, l, k', l'. Let J be the set of edges joining v to nodes of K_1. Let $j \in J$. Then another comb inequality is

$$x_k + x_l + x_{k'} + x_{l'} + x_j - x(J \backslash \{j\}) \leq 4. \tag{16}$$

(Note that this is easily seen to be valid—every hamiltonian cycle must contain an even number of edges of $\delta(V(K_1))$. The only way that (16) could be violated would be if all of $\{j, k, l, k', l'\}$ were used, but in this case a sixth edge, having coefficient -1 in (16) would be used, so (16) would still hold.)

Finally, we have one new type of inequality. This is the following *island constraint*. Let v, k, l, k', l', J, K_0, K_1, and K_2 be as above. The island constraint is

$$x(J) + x_{k'} + x_{l'} - x_k - x_l \geq 0. \tag{17}$$

Again, the validity is easily seen. Every hamiltonian cycle contains an even number of edges $\delta(V(K_1))$. Since all edges of $\delta(V(K_1))$ have coefficients ± 1 in (17), the only way we could violate (17) would be to have some hamiltonian cycle contain only k and l from $\delta(V(K_1))$. But in this case, it would contain no nodes of K_0, a contradiction.

(This is a special case of an inequality valid for the traveling salesman polytope of any graph containing a 2-1-cut $\{k, l, v\}$. Let K be a component of $G \backslash \{k, l, v\}$. Let u_1 and u_2 be the end nodes of k and l belonging to K. Let S be any proper subset of the nodes of K which includes u_1 and u_2. Then every hamiltonian cycle must contain an even number, at least two, of edges $\delta(S)$. But no hamiltonian cycle can use just k and l. Therefore, the inequality

$$x(\delta(S) \backslash \{k, l\}) - x_k - x_l \geq 0$$

is valid for the traveling salesman polytope.)

We now deal with the case that no 1-2-cuts exist.

THEOREM 4.1. *Let $G = (V, E)$ be an outer-facial graph. Let H be the associated Halin graph, and let B be the set of new edges introduced. Let $B' \subseteq B$ be a set of edges contracted, resulting in an outer-facial graph $\bar{G} = (\bar{V}, \bar{E})$. Furthermore, suppose that $\bar{G}$ has no* 1-2-*cuts. Then the international salesman polytope of $\bar{G}$ is given by the system* (7)–(11), *plus all equations* (13) *plus all inequalities* (14)–(17).

Proof. We proceed by induction on $|B'|$. If $|B'| = 0$ then the result is just Corollary 3.4. Choose $j \in B'$, and let $G' = (V', E')$ be obtained from G by contracting all edges in $B' \backslash \{j\}$. Then G' has no 1-2-cuts, and $\bar{G}$ is obtained from G' by contracting j. By induction, the international salesman polytope for G' is given by (7)–(11) and (13)–(17).

Since $j \in B$, it must belong to a 3-edge cut of G, and since $\bar{G}$ has no 1-2-cuts, j cannot belong to any 2-1-cuts of G'. Therefore, there is an equation (8) involving j, namely,

$$x_j + x_k + x_l = 2 \tag{*}$$

where k, l are the other two edges of a 3-edge cut containing j and both k and l are in G'. We will now proceed to use this equation to eliminate all other occurrences of x_j from any of the other inequalities (7)–(11) and (13)–(17). Thereby we obtain a linear system sufficient to define the international salesman polytope for $\bar{G}$. Let w be the node of $\bar{G}$ obtained when we contract j.

Case 1. $x(S) = 2$, where S is a 3-edge cut containing j. Then subtracting (*) we obtain an equation of the form (13).

Case 2. $x(\delta(v) \backslash \{h\}) - x_h \geq 0$, where $h \in \delta(v)$.

There are two subcases.

Case 2a. $h = j$. Adding the equation (*) yields $x(\delta(v) \backslash \{h\}) + x_k + x_l \geq 2$, which is a subtour constraint (14).

Case 2b. $j \in \delta(v) \backslash \{h\}$. Subtracting (*) we obtain

$$x(\delta(v) \backslash \{h, j\}) - x_h - x_k - x_l \geq -2$$

which is a comb inequality (15).

Case 3. $x(S - h) - x_h \geq 0$ for a 3-edge cut $S = \{h, k', l'\}$ of G.

There are two subcases, depending on whether $j = h$.

Case 3a. If $j = h$, we add the equation (*) and add the equation (13) if $\{k, l\} \neq \{k', l'\}$ to obtain the equivalent inequality

$$2x_k + 2x_l \geq 2$$

or

$$x_k + x_l \geq 1. \tag{18}$$

These inequalities are implied as follows. Add the two subtour elimination constraints (14) involving w, k, and l in $\bar{G}$. Then subtract the degree constraint (7) for w. Dividing by two gives (18). (See Fig. 4.4.)

Case 3b. $j \neq h$. Subtract equation (*) and add equation (13) if $\{k, l\} \neq \{k', l'\}$. We obtain the equivalent inequality $2x_h \leq 2$, which we have already seen is implied.

Note that edge $j \in B$ cannot belong to the co-boundary of a country or a city-state, so we do not have to consider (7) or (11). In addition, since contracting j does not result in a 1-2-cut, x_j cannot be present with a nonzero coefficient in equation (13) for G'.

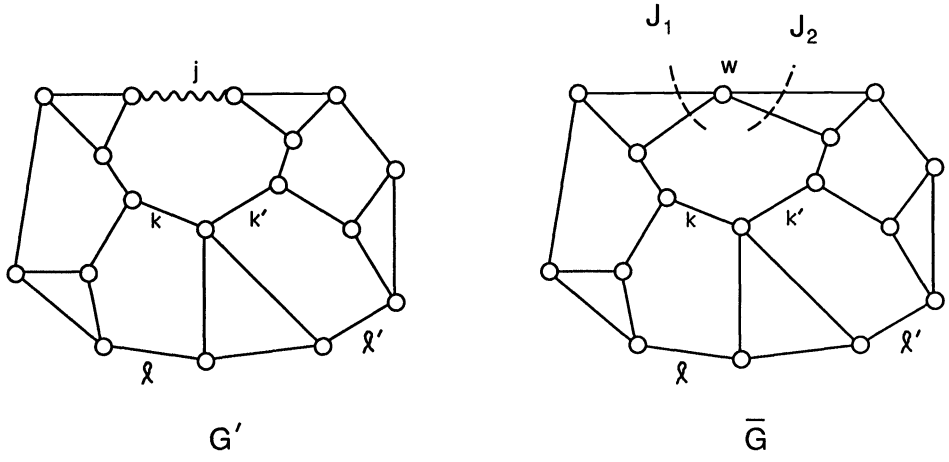


FIG. 4.4

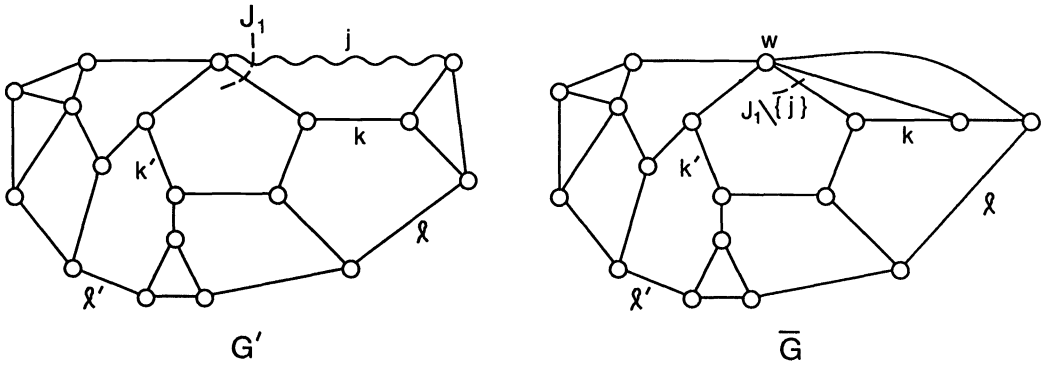


FIG. 4.5

Case 4. Subtour elimination constraint $x(J_1) + x_{k'} + x_{l'} \geq 2$, where $j \in J_1$. (See Fig. 4.5.)

Subtracting (*) gives an island constraint (17). Note that since contracting j does not result in a 1-2-cut, j is neither k' nor l'.

Case 5. Comb inequality (15) $x(J_1 - e) - x_e - x_{k'} - x_{l'} \geq -2$. If $j = e$, then adding (*) we obtain an island constraint (17). If $j \neq e$ (see Fig. 4.6) then we subtract (*) and obtain the inequality

$$x(J_1 \backslash \{e, j\}) - x_e - x_{k'} - x_{l'} - x_k - x_l \geq -4,$$

or

$$x_e + x_{k'} + x_{l'} + x_k + x_l - x(J_1 \backslash \{e, j\}) \leq 4.$$

This is a comb inequality (16).

Finally, note that for both the comb inequality (16) and the island constraint (17), if j occurred with a nonzero coefficient, then $\bar{G}$, obtained from G' by contracting j, would have a 1-2-cut, a contradiction. Therefore, x_j is not present with a nonzero coefficient in either of these constraints.

This completes the proof of Theorem 4.1. □

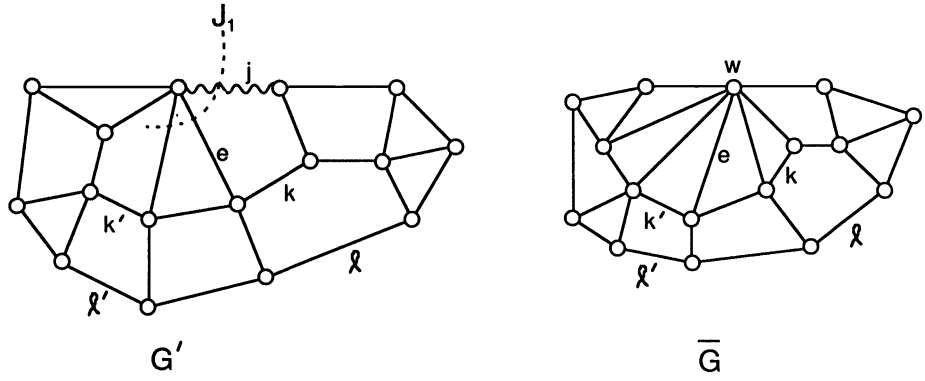


FIG. 4.6

COROLLARY 4.2. *Let G be an outer-facial graph with no* 1*-*2*-cuts. Then the* TSP *polytope for G is given by the system* (7)–(11) *and* (13)–(17).

5. Concluding remarks. Suppose G is an outer-facial graph and H is the associated Halin graph. The problem of finding a minimum weight feasible cycle in H is a special case of a more general problem that we now define.

Let G be a (general) graph with edge weights and let $\{V_1, \ldots, V_t\}$ be a collection of subsets of the node set V such that every node occurs in at least one of the subsets. If one thinks of the node set V as representing *cities*, then the sets V_i may be thought of as *countries*; and those cities that occur in more than one country may be thought of as *border cities* (e.g., Berlin before unification). Let C be a cycle in G. If C contains a city of G that occurs in only one country, then we say that C *visits* this country. If C contains a city that occurs in more than one country, then we may choose any nonempty subset of these countries and also say that C *visits* these countries. C is called *feasible* if we can choose the countries it visits so that every country is visited and the cities visited in any country induce a path of C. In other words, a feasible cycle of G can be interpreted as a tour which visits each country exactly once. The ISP is the problem of finding of feasible cycle of minimum weight. Observe that the TSP is a special case of an ISP where each node is itself a single country.

The polyhedral results of §3 of this paper address an ISP problem. These ISPs occur on Halin graphs, but are more general than TSPs on these graphs.

Appendix. In this section we briefly address the algorithmic problems of recognizing outer-facial graphs and finding a hamiltonian cycle in them.

We can determine whether a given graph $G = (V, E)$ is outer-facial in linear time as follows. First, we check for 3-connectivity and embed the graph in the plane (arbitrarily) in linear time (see Hopcroft and Tarjan [9] and [10]). We now must determine whether some face shares a vertex with every other face. Günter Rote observed that, for any face F, the number of different faces which share a vertex with F is precisely the sum of the degrees of the nodes of F, minus twice the number of nodes of F. The number of face-node incidences is twice the number of edges, and so these sums can be computed in linear time. Since we know the total number of faces, we can identify immediately the suitable outer face, if it exists.

One can devise an algorithm for finding a hamiltonian cycle in an outer-facial graph using dynamic programming. A general approach for constructing such an algorithm for classes of graphs that include outer-facial graphs is described in [1]. More specifically, one can use the basic idea from [3] that worked for Halin graphs. In this approach the algorithm takes as input an outer-facial graph G and then forms $H(G)$. The algorithm proceeds by shrinking fans until

a wheel is obtained. When we shrink a fan in $\tilde{H}$, we update the costs in the new graph in such a way that a minimum cost cycle $\tilde{C}$ satisfying (i)–(iii) of Proposition 2.1 will have the same cost as a minimum cost member of $\mathcal{P}$ in H, and hence as a minimum cost hamiltonian cycle in G. The algorithm proceeds by shrinking fans and updating costs until a wheel is found at which time a minimum cost member of $\tilde{C}$ is obtained directly. The corresponding hamiltonian cycle in G can be obtained by reversing the shrinking process. Such dynamic programming algorithms have linear worst case time complexity.

This algorithm is described in detail in [8]. One interesting feature is that, as it proceeds, it always considers an ISP of the same form as the original, in a smaller graph with modified costs.

REFERENCES

[1] M. W. Bern, E. L. Lawler, A. L. Wong, *Why certain subgraph computations require only linear time*, Proc. 26th Ann. IEEE Symp. Found. Comp. Sci., 1985, pp. 117–125.

[2] V. Chvátal, *Linear Programming*, W. H. Freeman and Company, New York, 1983.

[3] G. Cornuejols, D. Naddef, and W. R. Pulleyblank, *The travelling salesman problem in graphs with 3-edge cutsets*, J. Assoc. Comput. Mach., 32 (1985), pp. 383–410.

[4] C. R. Coullard and W. R. Pulleyblank, *On cycle cones and polyhedra*, Linear Algebra Appl., 114/15 (1989), pp. 613–640.

[5] J. B. J. Fourier, *Second extrait*, in *Oeuvres*, G. Darboux, ed., Gauthiers-Villars, Paris, 1890, pp. 325–328.

[6] M. Grotschel and M. W. Padberg, *On the symmetric travelling salesman problem* I: *Inequalities*, Math. Programming, 16 (1979), pp. 265–280.

[7] ———, *On the symmetric travelling salesman problem* II: *Lifting theorems and facets*, Math. Programming, 16 (1979), pp. 281–302.

[8] D. Hartvigsen and W. R. Pulleyblank, *Outer-facial graphs and the traveling salesman problem: Algorithm and polyhedron*. Research Report, Center for Research in Business, University of Notre Dame, Notre Dame, IN, 1994.

[9] J. E. Hopcroft and R. E. Tarjan, *Dividing a graph into triconnected components*, SIAM J. Comput., 2 (1973), pp. 135–158.

[10] J. E. Hopcroft and R. E. Tarjan, *Efficient planarity testing*, J. Assoc. Comput. Mach., 21 (1974), pp. 549–568.

[11] D. A. Kohler, *Translation of a report by Fourier on his work on linear inequalities*, Opsearch, 10 (1973), pp. 38–42.

[12] M. Padberg and G. Rinaldi, *A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems*, SIAM Rev., 33 (1991), pp. 60–100.

SIAM J. OPTIMIZATION
Vol. 4, No. 4, pp. 691–707, November 1994
© 1994 Society for Industrial and Applied Mathematics

LARGE-SCALE, NONLINEARLY CONSTRAINED OPTIMIZATION ON A 1024-PROCESSOR nCUBE*

J. H. GLICK† AND J. B. ROSEN‡

Abstract. An efficient parallel partition algorithm is described that has been developed for solving a class of large-scale, structured, nonlinear optimization problems having many nonlinear constraints. This class of problems has an objective function and constraints that are linear with respect to a large number of variables and nonlinear with respect to a relatively small number of other variables. Also, the Jacobian of the constraint matrix with respect to the linear variables is block diagonal and dense with respect to the nonlinear variables, so the nonlinear variables are linking variables in an otherwise separable problem. Problems having this structure arise in applications that include combined manufacturing and distribution modeling and stochastic optimization.

The partition algorithm first projects the original large-scale problem onto the much smaller space of the nonlinear linking variables. A serial version was implemented on a Cray X-MP and a parallel version on a 1024-processor nCUBE multicomputer. The algorithm was tested on a range of quadratic constraint test problems, the largest having over one million quadratic constraints. The authors' studies showed that serial implementation gives good solution times when compared to using a general purpose nonlinear programming package or a bundle method. The parallel implementation was able to solve the very large problems with good efficiency.

Key words. nonlinear programming, large-scale optimization, parallel optimization

AMS subject classifications. 90C30, 90C06

1. Introduction.

Many large-scale, constrained, nonlinear programming problems have certain common features; the constraint matrix is structured and sparse, and many of the variables are linear. Problems of this general type occur in many important large-scale optimization applications, including stochastic optimization [3], combined manufacturing and distribution modeling [8], and structural design optimization [15]. Such problems may also have many nonlinear constraints. In this paper we summarize our investigations into using a new partition algorithm to solve problems of this class that have a specific structure. The problems used for the computational testing are based on a typical two-stage stochastic optimization model [3] in which the number of blocks is determined by the number of possible scenarios considered. This is described in §4. Our studies show that a serial version, implemented on a Cray X-MP computer, gives good solution times when compared to two alternate algorithms, and that a parallel version, implemented on a 1024-processor nCUBE 2 multicomputer, solves very large problems with good efficiency. The results presented here represent a part of the Ph.D. thesis research of Glick [4].

The problem we consider is given by

$$\begin{aligned} &\underset{y,w}{\text{maximize}} && f(y) + b^T w \\ &\text{subject to} && A^T w \leq c(y), \end{aligned} \tag{1}$$

* Received by the editors September 7, 1993; accepted for publication (in revised form) February 11, 1994. This work was supported in part by Air Force Office of Scientific Research grant AFOSR-91-0147, the Minnesota Supercomputer Institute, the Army High Performance Computing Research Center, and Sandia National Laboratories.

† Department of Computer Science, University of Minnesota, Minneapolis, Minnesota 55455. Present address, Department of Mathematics and Computer Science, University of San Diego, San Diego, California 92110 (glick@teetot.acusd.edu).

‡ Department of Computer Science, University of Minnesota, Minneapolis, Minnesota 55455, and Department of Computer Science and Engineering, University of California, San Diego, La Jolla, California 92093.

where $y \in \Re^s$, $w, b \in \Re^m$, $A \in \Re^{m \times n}$ is block diagonal with k blocks, $f : \Re^s \to \Re$, and $c : \Re^s \to \Re^n$. We make the following assumptions about the problem.

A1. $f(y)$ and $c_i(y)$, $i = 1, \ldots, n$ are concave functions. This implies that (1) is a convex programming problem.

A2. $f(y)$ and $c_i(y)$, $i = 1, \ldots, n$ are twice differentiable functions.

A3. Rank$(A) = m$, i.e., A has full rank.

A4. A solution (y^*, w^*) to (1) exists.

The number s of linking variables (the vector y) is typically much smaller than the number m of linear variables. The total number n of inequality constraints is assumed to be greater than m. With the block diagonal structure assumed for A, the problem constraints can be written

$$\begin{pmatrix} A^1 & & \\ & \cdot & \\ & & A^k \end{pmatrix}^T \begin{pmatrix} w^1 \\ \cdot \\ w^k \end{pmatrix} \leq \begin{pmatrix} c^1(y) \\ \cdot \\ c^k(y) \end{pmatrix}. \tag{2}$$

Here, vectors and matrices associated with individual blocks are denoted by superscripts. It should be noted that assumption A4 is included to simplify the discussion of the algorithm. In practice, an unbounded problem would be recognized by the algorithm that we present. Finally, although we do not explicitly include simple bounds on the variables in the problem statement, they can be treated in a straightforward manner, as is discussed in §2.

A property of the problem (1) is that for fixed values of the nonlinear linking variables y it separates into k smaller problems, one for each constraint block. The partition algorithm takes advantage of this property by projecting it onto the space of the y variables. This transformation gives the problem

$$\underset{y \in Y}{\text{maximize}}\ \Psi(y), \tag{3}$$

where

$$\Psi(y) = f(y) + \Phi(y), \tag{4}$$

$$\Phi(y) = \max_w \{b^T w \mid A^T w \leq c(y)\}, \tag{5}$$

and

$$Y = \{y \mid \text{there exists a } w \text{ such that } A^T w \leq c(y)\}. \tag{6}$$

The advantages of this formulation are that it usually has a much smaller dimension (s versus $m + s$) than (1), and the objective function decouples into k relatively small linear programs (one for each block of the matrix A) that can be solved in parallel. The disadvantages are that it is a nonsmooth problem and the constraint set is not defined explicitly. Properties of the projected problem are discussed in [4, Chap. 4].

We have concentrated our study on algorithms for solving the projected problem (3) that are rich in block-level parallelism; that is, they contain many tasks that separate into subtasks associated with the blocks of the constraint matrix A. This includes, but is not limited to, solving for $\Phi(y)$ in (5). This coarse-grained parallelism is well suited for distributed memory multiple instruction multiple data (MIMD) parallel computers, so this is the architecture that we used for the parallel implementation of the partition algorithm.

The partition approach to solving projected problems in large-scale optimization was first proposed by Rosen [20]–[22]. When applied to (3), it considers the feasible set Y as consisting of contiguous regions that are delineated by the surfaces of gradient discontinuities in the objective function. For each region, a smooth problem, called the reduced problem, can be formulated that is equivalent to (3) within its bounds. Partition algorithms solve, or approximately solve, a sequence of these reduced problems in such a way that they converge to the solution of (1). The reduced problems are smooth, so efficient nonlinear optimization techniques can be used to solve them, but care must be taken because they have nonconvex feasible sets.

We call the partition algorithm described in §3 the modified partition algorithm because it evolved from an earlier partition algorithm (the basic partition algorithm) [5], [4]. We preface our presentation of the modified partition algorithm with a brief description of the basic partition algorithm in §2.

We implemented a serial version of the modified partition algorithm on a Cray X-MP computer and tested it on a series of randomly generated test problems. The largest problem attempted, with 10 nonlinear variables, 512 blocks, 10,240 constraints, and 5120 linear variables, was solved in 54 seconds. The observed time complexity with respect to the number of blocks was linear. We implemented a parallel version on a 1024-processor nCUBE 2 multicomputer. The largest problem that would fit into the combined memory of all 1024 processors had 10 nonlinear variables, 65,536 blocks, 1,310,720 constraints, and 655,360 linear variables, and was solved in 136 seconds with an efficiency of approximately 0.5. We also found the scaled speedup curve to be close to linear, which suggests the algorithm is well scalable and could efficiently use more processors. Our test problem generator is described in Appendix A, and computational results are presented in §4.

2. The basic partition algorithm. In this section we give a summary of our earlier basic partition algorithm [4, Chap. 5], from which the modified partition algorithm evolved.

The basic partition algorithm for solving the projected problem (3) begins with a feasible point $\overline{y} \in Y$, where the objective function $\Psi(\overline{y})$ is evaluated using (4) and (5). Corresponding to the solution of the linear program (5) are a vector $x \in \Re^n$ of Lagrange multipliers and m basic constraints. Using the optimal basis of (5), the matrix A, the vector $c(y)$, and the Lagrange multiplier vector x are partitioned into basic and nonbasic parts:

$$\begin{aligned} &\hat{A} = [B \;\; N], \\ &\hat{c}(y) = [c_B(y)^T \;\; c_N(y)^T]^T, \quad \text{and} \\ &\hat{x} = [x_B^T \;\; x_N^T]^T. \end{aligned} \tag{7}$$

Also defined are I_B, the set of basic constraints, and I_N, the set of nonbasic constraints. Using this partitioning, the constraints of the original problem (1) are written as

$$\begin{aligned} &B^T w \le c_B(y) \quad \text{and} \\ &N^T w \le c_N(y). \end{aligned} \tag{8}$$

At $\overline{y}$, the first set of these constraints is satisfied exactly and the second set is satisfied, but is not necessarily binding. The relation $B^T w = c_B(\overline{y})$ is used to eliminate w from

(1), giving the following reduced problem in y-space.

$$\begin{array}{ll} \underset{y}{\text{maximize}} & \Gamma(y) \\ \text{subject to} & \pi_i(y) \leq 0 \quad \text{for } i \in I_N, \end{array} \tag{9}$$

where

$$\begin{aligned} &\Gamma(y) = f(y) + (B^{-1}b)^T c_B(y) \quad \text{and} \\ &\bar{\pi}(y) = (B^{-1}A)^T c_B(y) - c(y). \end{aligned} \tag{10}$$

It can be shown [4, Theorem 5.1] that the feasible set of this problem is also the region of Y in which B is an optimal basis of the linear program (5), so within its feasible set, (9) is equivalent to the projected problem (3). It can also be shown [4, Lemma 5.1] that Y is covered by regions, each being the feasible set of a reduced problem corresponding to a different basis of (5). Finally, it should be noted that the feasible set of the reduced problem is not necessarily convex, because nothing can be said about the convexity of $\pi(y)$ in (10).

The basic partition algorithm forms the reduced problem at the point $\bar{y}$ and finds a Karush–Kuhn–Tucker (KKT) point of it. This KKT point may also be the solution of (3), and the partition algorithm tests for this by computing estimates of the Lagrange multipliers of the basic constraints from

$$x_B = u_B - B^{-1} \sum_{i \in I_N} x_i a_i, \tag{11}$$

where u_B are the Lagrange multipliers at the solution to the linear program (5) and a_i is the ith column of A. If these multipliers are nonnegative, then the KKT point solves (3) (see [4, Thm. 5.2]), and the algorithm terminates. If $x_B \not\geq 0$, then the basis B is changed in such a way that a KKT point of the reduced problem corresponding to the new basis will have an improved function value. This change of basis is accomplished by solving the linear program

$$\begin{array}{ll} \underset{x_B, x_i, i \in I_S}{\text{minimize}} & x_p \\ \text{subject to} & Bx_B + \displaystyle\sum_{i \in I_S} a_i x_i = b, \\ & x_B \geq 0, \quad x_i \geq 0 \quad \text{for } i \in I_S. \end{array} \tag{12}$$

Here, I_S is the set of nonbasic constraints active at the reduced problem KKT point, and p is the index of the most negative basic multiplier. Solving this problem replaces a basic constraint having a negative multiplier estimate with a nonbasic constraint having a positive multiplier estimate.

Using the resulting new basis, the basic partition algorithm repeats the process of forming a reduced problem, finding a KKT of it, and testing for optimality of the original problem. Under a mild no-cycling assumption, it can be shown [4, Thm. 5.4] that this process will find a solution to (1) in a finite number of iterations.

Clearly, an important part of the basic partition algorithm is finding a KKT point of the reduced problem (9). Because the feasible set of the reduced problem is nonconvex, some nonlinear programming methods have difficulty with this task. We chose to solve it using a linearization algorithm proposed by Pshenichny [19, p. 188], and modified by Belegundu and Arora [2] (see also [14], [1]). We have also made several modifications of the algorithm that improve its efficiency. There are two

principle reasons why we have chosen to use this particular algorithm. First, it has proven to be effective for problems with nonconvex feasible sets. The convergence of Pshenichny's algorithm does not rely on a convex feasible set, and in fact, it was designed to solve problems such as (9). Second, it deals with the constraints in an efficient manner. Although the dimension of the reduced problem is just s, the number of constraints can be quite large $(n - m)$, and it is important to deal with them efficiently. Pshenichny's algorithm does this by only considering a subset of the constraints as being active for the purposes of computing a search direction. By doing this, the size of the quadratic programming problem that determines the search direction is kept small.

The two principle steps of Pshenichny's algorithm are solving a direction-finding problem and executing a linesearch. The direction-finding step uses an active set approach to limit the number of constraints that are included. In this strategy, only those constraints that are within some tolerance of being active are used in the computation of the search direction. The direction-finding problem is given by

$$\begin{aligned} &\underset{d\in\Re^s}{\text{maximize}} && \nabla\Gamma(y)d - \tfrac{1}{2}d^T G(y)d \\ &\text{subject to} && \pi_i(y) + \nabla\pi_i(y)d \le 0 \quad \text{for } i \in J(\delta, y), \end{aligned} \tag{13}$$

where $G(y)$ is a positive definite approximation to the Hessian of the Lagrangian $H(y)$ of (9) and $J(\delta, y)$ is the set of the nonbasic constraints that are within δ of being active. In our implementation of Pshenichny's algorithm, δ is dynamically adjusted to try to include in the direction-finding problem only those constraints that will affect the resulting direction. This direction-finding problem is similar to those of SQP algorithms in nonlinear programming.

The computed search direction d is first used in the following test to determine if the current point is a KKT point of the reduced problem

$$\|G(y)d\|_\infty \le k\,\text{tol}\,\|x\|_1. \tag{14}$$

$G(y)d$ is equivalent to the residual of the projected gradient optimality condition of the reduced problem and tol is the optimality tolerance. The tolerance is scaled by the norm of x, the current multiplier estimate of the original problem (1), and also by the number of blocks k. We found it necessary to scale the tolerance by the number of blocks to account for the loss of precision in summing certain quantities across all the blocks. This was only necessary for problems having very many blocks (> 4096), but we scaled the tolerance by k for all test problems.

If the termination criterion (14) is not satisfied, then a linesearch is performed that approximately maximizes an absolute penalty function, from which point the next iteration begins.

We conclude this section with a discussion of how simple bounds on the variables of (1) could be efficiently handled. Bounds on w would also appear as simple bounds on the k linear programming problems (5). Because we assume that $n > m$, it is more efficient to solve the primal form of (5), where the bounds on w become additional columns in the constraint matrix, which does not increase the size of the basis. Bounds on y should not be handled in the context of the general constraints $A^T \le c(y)$, but rather should be carried through directly to the reduced problem (9), where they would be treated by Pshenichny's algorithm in the same manner as the other constraints.

Details of the basic partition algorithm, along with computational results and analysis, can be found in [4, Chap. 5].

3. The modified partition algorithm. In this section we describe the modified partition algorithm for solving the projected problem (3). The motivation for it comes from two observations about the basic partition algorithm described in §2. First, it seems inefficient to compute a KKT point of a reduced problem that may be far from the solution to the projected problem, and second, it is not well-suited to block parallelization as the time-consuming work does not easily separate into tasks associated with blocks. Most notably, it does not spend time evaluating $\Psi(y)$, a task that does separate into independent subproblems.

We have addressed each of these observations in modifying the basic partition algorithm. First, we only approximately solve each of the reduced problems that are formed, and second, the approximate solution to a reduced problem is used as a direction for a linesearch on the projected problem objective function $\Psi(y)$. Our results show that a significant fraction of time in the algorithm is spent evaluating $\Psi(y)$ or performing other block-separable tasks, and so is well parallelizable.

The modified partition algorithm begins with a feasible y value, and each iteration starts with the identification of the optimal basis associated with the current y value. The optimal basis for the first iteration is identified by solving for $\Psi(y)$; for subsequent iterations it will have been determined during the previous iteration. The reduced problem corresponding to this basis is formed, and the set of its active constraints is identified. A constraint is considered active if it is within some value δ of being satisfied exactly, where δ is a dynamically adjusted parameter. A quadratic approximation to the current reduced problem is then constructed, including only those constraints that are active. The objective function of the quadratic program is a quadratic approximation to the Lagrangian of the reduced problem, and the constraints are linearizations of the active constraints (but they remain inequality constraints). This quadratic program is identical to the direction finding problem (13) of the reduced problem solver described in §2.

The solution to this problem is first used to test if the current solution estimate is a KKT point of the reduced problem. If it is, and if the updated Lagrange multiplier estimates of the basic constraints are nonnegative, then the solution to (3) has been found and the algorithm terminates. If the current point is a KKT point of the reduced problem, but one or more of the Lagrange multipliers are negative, the basis is modified by solving the linear program (12) as described in the basic partition algorithm.

If the current point is not a reduced problem KKT point, then the solution to the quadratic program (13) is used as the direction of a linesearch on the projected problem objective function $\Psi(y)$. We call this a global linesearch; it searches for a trial point that gives sufficient improvement in $\Psi(y)$ (as determined by the Armijo rule) and has a stepsize of at least one-half. If successful, the trial point is accepted as the next approximation to the solution.

If the global linesearch is unsuccessful, the algorithm reverts to the strategy of the basic partition algorithm; that is, it finds a KKT point of the current reduced problem, tests for optimality of (3), and, if not optimal, modifies the basis to give a different reduced problem. The algorithm then returns to its global linesearch strategy.

A summary of the modified partition algorithm is given below. Details of it and a proof of its convergence are given in [4, §6.3]. The two key elements of the convergence proof are (i) showing that each global linesearch terminates after a finite number of function evaluations, and (ii) showing that during a sequence of global linesearches, the algorithm either converges to a reduced problem KKT point or terminates un-

successfully after a finite number of linesearches and switches to the basic partition strategy.

ALGORITHM 3.1 (MODIFIED PARTITION ALGORITHM).

Begin

1. *Initialize.*
   Solve the linear program (5) at an initial feasible point y to get the starting basis B.
2. *Solve the direction-finding problem.*
   Form the reduced problem associated with the current basis. Form and solve the direction-finding problem (13) for this reduced problem at the current point, getting a direction d and multipliers x_i, for $i \in J(\delta, y)$.
3. *Compute the multiplier estimates.*
   Set $x_i = 0$ for $i \in I_N \setminus J(\delta, y)$, and compute x_B, the Lagrange multiplier estimates for the basic constraints, from (11).
4. *Test for optimality.*
   Use (14) to test if the current point is a KKT point of the reduced problem. If it is, and if $x_B \geq 0$, then terminate with the current point as the solution. If the current point is a KKT point of the reduced problem, but $x_B \not\geq 0$, modify the basis by solving the linear program (12) and go to step 2. If the current point is not a KKT point of the reduced problem, continue to step 5.
5. *Perform a global linesearch.*
   Perform a global linesearch in the direction d. If a point is found that gives a sufficient increase in the function value $\Psi(y)$ and that has a steplength of at least one-half, accept this point as the new solution estimate. The solution of (5) at this new point identifies the basis to be used in the next iteration. Go to step 2. If the global linesearch is unsuccessful, use the reduced problem solver of the basic partition algorithm to find a KKT point of the current reduced problem. Test for optimality of (3) as in step 4 and terminate with the KKT point as the solution if it is optimal. Otherwise, modify the basis by solving the linear program (12) and go to step 2.

End

In the parallel version of the modified partition algorithm, each processor independently runs its own copy of Algorithm 3.1. The k problem blocks are distributed equally among the processors, and when a task is encountered that is block separable, each processor performs it for only those blocks it has been assigned. There may be overhead in each block-separable task; overhead may be due to interprocessor communication, processor idle time, or nonuseful work. Nonuseful work is work done by the parallel algorithm that would not be performed by the serial algorithm in solving the same problem. The principal block separable task is the solution of the linear program (5), but there are others. A complete listing of them is found in [4, Chap. 6].

4. Computational results. In this section we present the results of our computational testing of the modified partition algorithm on a set of generated test problems. The choice of test problems was motivated by the desire to measure the performance of the algorithms (both sequential and parallel) as the number of blocks increased. A two-stage stochastic optimization model permits the construction of test problems for which this can be done.

In the first stage of the stochastic model, the vector $y \in \Re^s$ is fixed. This vector may, for example, represent plant design parameters that occur in a nonlinear manner

in the plant model. The objective function value associated with choosing a specific vector y is $f(y)$. In the second stage (at a later time) one of k possible scenarios occurs. The ith scenario is represented by the linear programming subproblem

$$\begin{array}{ll} \underset{w^i}{\text{maximize}} & (b^i)^T w^i \\ \text{subject to} & (A^i)^T w^i \leq c^i(y). \end{array} \tag{15}$$

The matrix A^i and vectors b^i and c^i determine the objective function and plant constraints corresponding to the occurrence of the ith scenario in the second stage. Once a specific scenario, say $i = j$, occurs, the optimal w^j is determined by solving (15) with $i = j$. The number of possible scenarios can grow rapidly with the subproblem size, so it is important to be able to handle very large values of k.

To determine the optimum vector y a probability p^i must be assigned to each scenario. The vector y is then determined by finding

$$\max_{y, w^i} f(y) + \sum_{i=1}^{k} p^i (b^i)^T w^i, \tag{16}$$

subject to the subproblem constraints (15). If we set

$$b^T = (p^1(b^1)^T, p^2(b^2)^T, \ldots, p^k(b^k)^T),$$

the problem is identical to that given by (1) and (2), with the number of blocks equal to the number of different scenarios assumed. The computational test problems were based on this model, with k as large as 65,536. Details of the random generation of these test problems are given in Appendix A.

For comparison, we also solved the test problems using the bundle method, an algorithm for nonsmooth optimization that we applied directly to the projected problem (3). The bundle method was developed by Lemarechal [9] and was later refined by him and others [6], [7], [10]–[13], [16], [17], [23]. The version that we implemented was developed by Kiwiel [6, Chap. 7] and incorporates several enhancements that improve the efficiency of the original algorithm. To show how a general purpose, nonlinear programming algorithm will perform in solving (1), we also solved some of the test problems using the MINOS (version 5.3) optimization package [18], which implements a projected augmented Lagrangian algorithm. It employs sparse matrix data handling techniques, so that unnecessary arithmetic associated with the zeros in the constraint Jacobian of the problem constraints is avoided, but it does not take advantage of the block-angular structure of the Jacobian. To the extent possible, we used equivalent termination criteria in all three algorithms for which we present results.

The primal simplex algorithm from MINOS, which also uses sparse matrix handling techniques, was used to solve the linear program (5) in primal form. The optimality tolerance used in both the linear program and reduced problem solvers was 10^{-6}.

Our primary interest is in the solution time complexity with respect to the number of problem blocks. This is of interest in stochastic programming applications where the number of blocks can be made arbitrarily large. Problem test sets 1 and 2 (in Appendix A) vary the number of blocks while holding other problem size parameters constant. In test set 1 the blocks in the constraint Jacobian are small and dense, while in test set 2 they are larger and sparse. We are also interested in the solution time complexity with respect to s, the number of linking variables. Test set 3 varies s

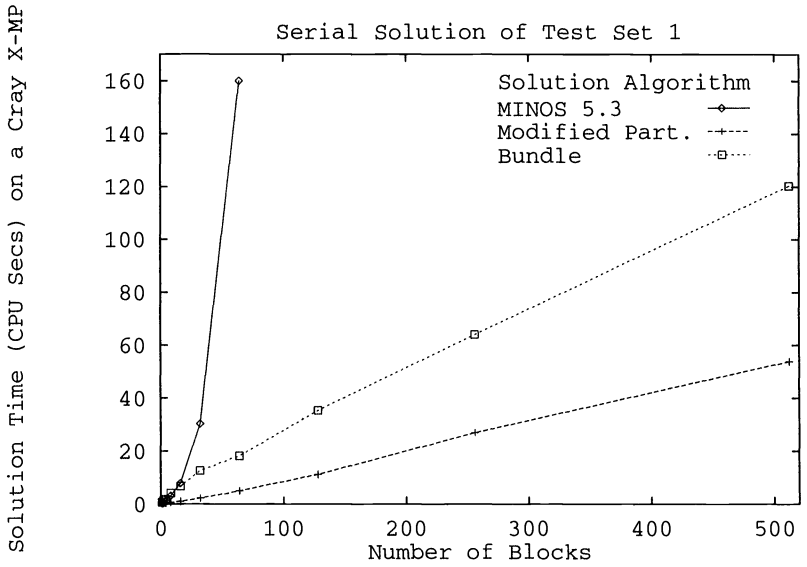


FIG. 1. *Performance of the algorithms on problems with dense blocks.*

for fixed values of the other parameters. The serial implementations of the algorithms were tested on all three test sets, while the parallel implementations were tested only on test set 1.

We first present results for the serial implementations on a Cray X-MP. Figure 1 shows the solution time versus the number of blocks for problems with dense blocks. For both the bundle method and the modified partition algorithm the observed complexity is nearly linear, with the constant of the modified partition algorithm being about half that of the bundle method. This complexity suggests that the algorithms could be used to solve much larger problems. In fact, our experiments with parallel implementations on an nCUBE 2 computer support this result for both the modified partition algorithm and the bundle method. In contrast, the time complexity of MINOS is close to quadratic. This is not surprising, as MINOS does not explicitly take advantage of the block structure of the Jacobian.

Both the modified partition algorithm and the bundle method are iterative algorithms, where each iteration consists of the solution of a direction-finding problem followed by a linesearch. Figure 2 compares the number of iterations taken by the modified partition algorithm and the bundle method on the problems of test set 1. After initial fluctuations, the number of iterations taken by both algorithms level off to values close to one another. This means that the time taken per iteration of the modified partition algorithm must be about half that of the bundle method. We found this to be due primarily to less time spent evaluating the projected problem objective function. Our experiments with the parallel versions of both algorithms show how the iteration count behaves for larger problems. Ideally, the iteration count for parallel and serial versions should be the same when solving identical problems, and in practice we found only slight differences. The iteration count decreased slowly as the number of blocks increased from 512 to 65,536, to ≈ 25 for the modified partition algorithm, and to ≈ 15 for the bundle method. We believe this decrease is due primarily to the

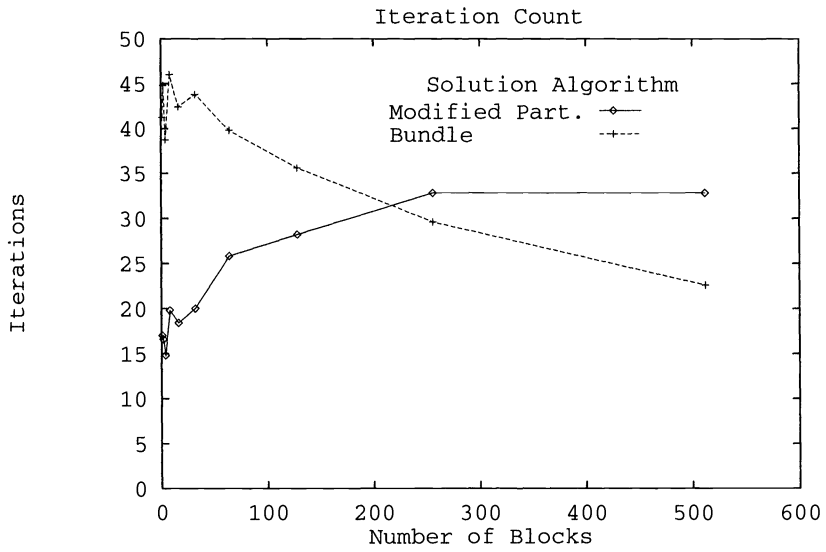


FIG. 2. *Number of iterations taken to solve problems with dense blocks.*

need to scale the termination criterion by the number of blocks for large problems. The percentage of iterations in which a global linesearch is taken, as compared to a linesearch in Pshenichny's algorithm, appears to vary with the number of blocks in the problem. For smaller problems, it is close to 50%, and it grows to nearly 90% for problems having 512 blocks or more. This is encouraging, as it seems that more rapid progress towards a solution is made during global linesearches.

It is interesting to compare how rapidly the algorithms approach the true answer. Figure 3 shows the relative error in the estimated optimal y value as a function of relative time. Here, a relative time of 1 is the time taken for the slowest algorithm, in this case the bundle method, to complete and relative error in y is defined as $\|y - y^*\|_2/\|y\|_2$. Figure 4 shows the relative error in the estimated optimal function value, also as a function of relative time, where relative error in the function value is defined as $|(\Psi(y) - \Psi(y^*))/\Psi(y^*)|$. In absolute terms, both $|\Psi(y) - \Psi(y^*)|$ and $\|y - y^*\|_2$ were less than 10^{-4} when the algorithm terminated. Both Figs. 3 and 4 represent data for a single problem, but they are representative of the behavior of the algorithms on all of the test problems. Both the bundle method and the modified partition algorithm make early rapid progress toward the solution, but whereas the modified partition algorithm continues this rapid progress to the solution, the progress of the bundle method slows, and it takes considerably longer to satisfy the termination criteria.

Figure 5 shows the solution time versus number of blocks for problems with a sparse blocks (test set 2). While the solution times are greater for the large sparse blocks than for the small dense blocks, the increase is most distinct for the bundle method. For example, the dense 256-block problems were solved in approximately 30 seconds by the modified partition algorithm and in approximately 65 seconds by the bundle method. The corresponding sparse 256-block problems were solved in approximately 140 seconds by the modified partition, while it is estimated the bundle

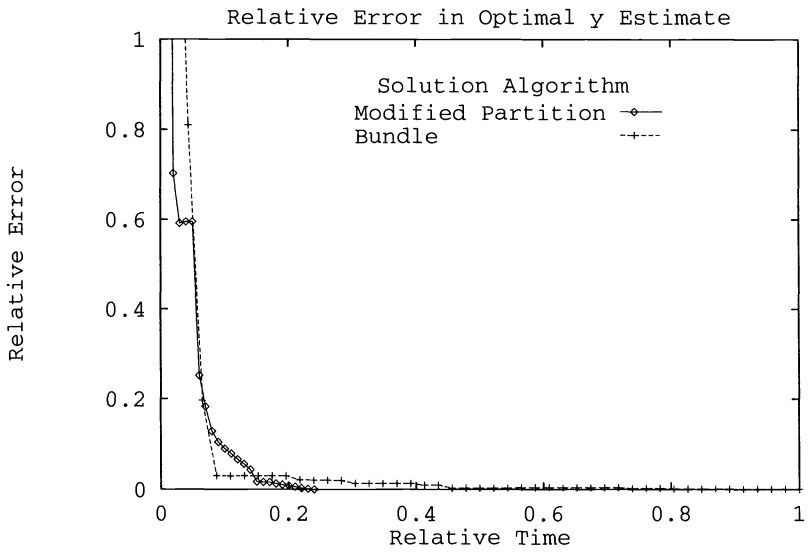


FIG. 3. *Relative error in the estimated optimal y value.*

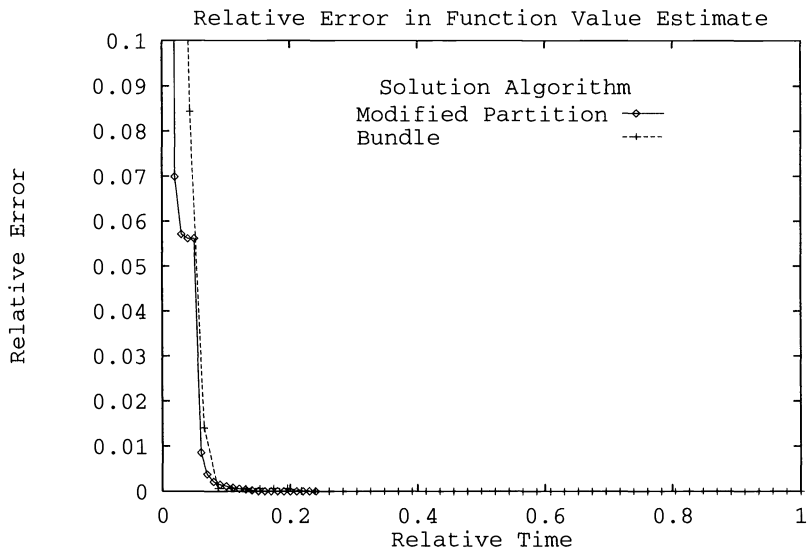


FIG. 4. *Relative error in the estimated optimal function value.*

method would require more than 1500 seconds. One reason for the increased solution times for the sparse block problems is an increase in the average time per iteration, primarily due to more time being required to evaluate the objective function of the projected problem. In fact, this is the principal source of the increased solution time for the modified partition algorithm. For the bundle method, however, there is also a marked increase in the number of iterations required to find a solution. This suggests

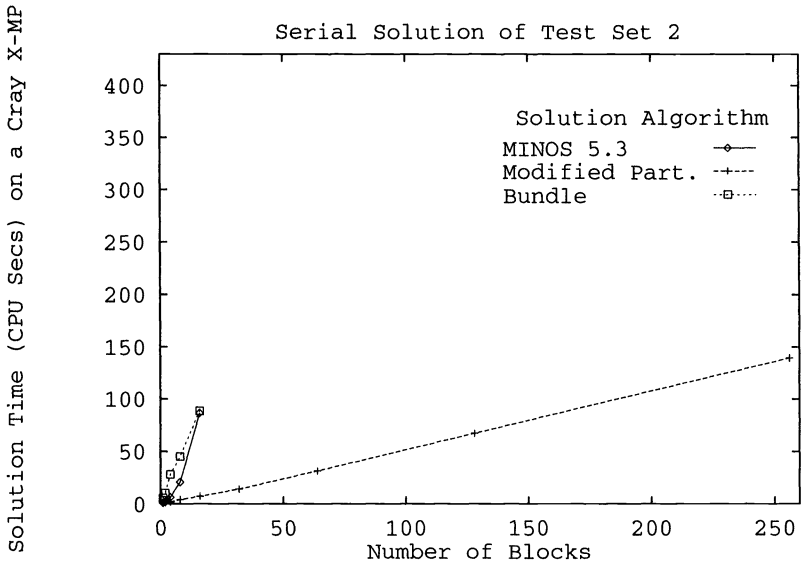


FIG. 5. *Performance of the algorithms on problems with sparse blocks.*

that for problems with large blocks in the constraints, the bundle method may not be a competitive solution method.

Figures 6 and 7 plot solution time versus s for several values of k on the problems of test set 3 for the modified partition algorithm and the bundle method, respectively. For both algorithms, the solution time depends approximately linearly on s with the curves intersecting the vertical axis above zero. This is consistent with the fact that while the work of some tasks depends linearly on s, other work (such as the evaluation of $\Psi(y)$) depends on k and is independent of s. Points on the curve that seem to indicate a greater than linear growth in solution time are due primarily to a greater number of major iterations required to solve the problem.

The possibility of cycling, caused by degeneracy, exists in two parts of the modified partition algorithm. The first part is during the solution of the linear programs (5), where cycling prevention rules can be employed. The second part is when the basis is modified by solving the linear program (12) when a KKT point to the current reduced problem is not the solution to the original problem. By definition, when the KKT point of a reduced problem is degenerate, the gradients of the active constraints are not linearly independent. When the basis is modified, then the new reduced problem will have at least one negative multiplier estimate and, if the current point is not degenerate, finding a KKT point would require moving away from the current point to one with an improved function value. If the point is degenerate, however, it is possible that the current point is a KKT point, and is recognized as such in the course of solving the new reduced problem. If this is in fact the case, then one or more iterations will have been taken without moving in y-space and without making any improvements in the function value. If a sequence of degenerate iterations occur in which a basis is repeated, then the algorithm begins to cycle. We ran the partition

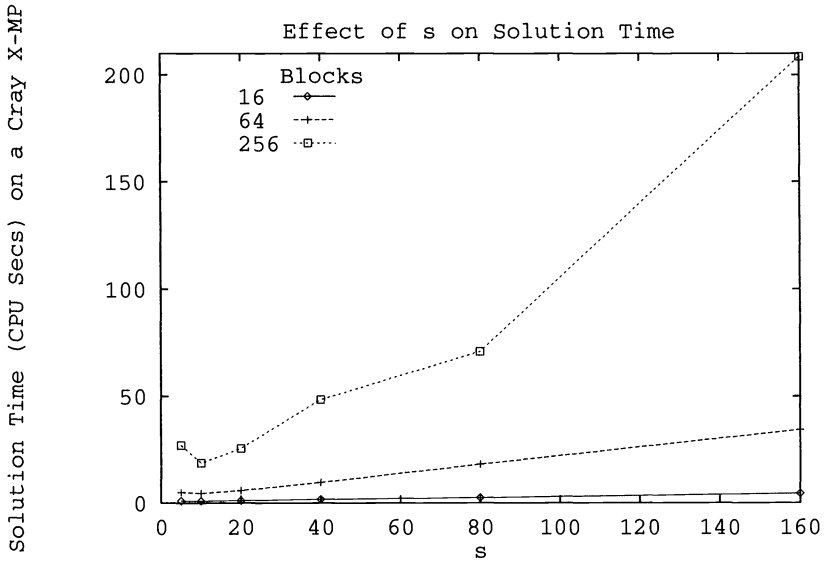


FIG. 6. *Dependence on s of the performance of the modified partition algorithm for several values of k.*

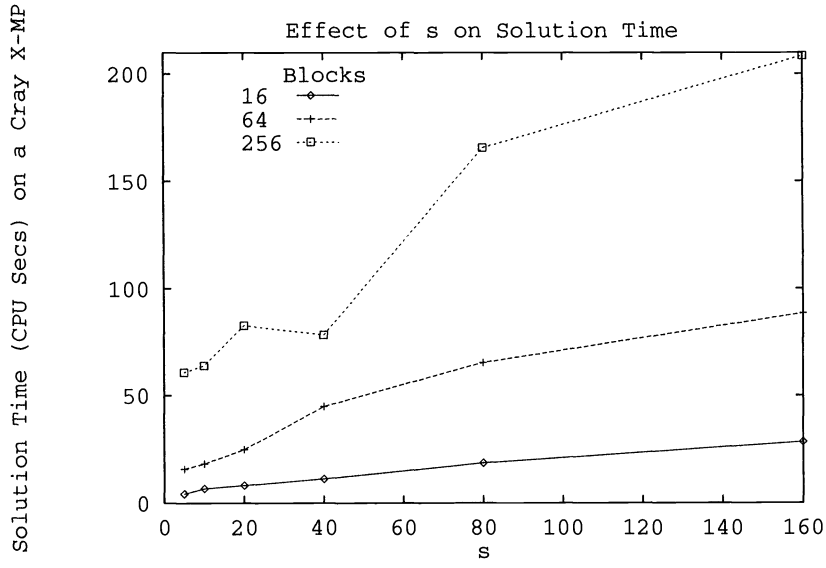


FIG. 7. *Dependence on s of the performance of the bundle method for several values of k.*

algorithm on a series of problems with highly degenerate solutions and in each case the solution was found, despite taking some degenerate steps. This suggests that while cycling is a possibility, in practice it is not observed often.

We now present results for the parallel implementations on a 1024-processor nCUBE 2 computer, which were used to solve the problems with dense blocks (test set 1). We present the performance data for both algorithms in two different ways: the execution time curve and the scaled speedup curve. In the execution time curve, we plot the elapsed time to solve the problem versus the number of blocks in the problem, where the number of processors is chosen to minimize the solution time. The scaled speedup curve plots speedup versus number of processors, where the number of blocks in the problem is chosen to fill the collective memory of the processors. For the problems of test set 1, 64 blocks can fit onto each processor.

The definition of speedup that we use is the ratio between the time it would take a single processor of the parallel machine to solve a problem using the serial version of the same algorithm to the actual time taken by the parallel machine to solve the problem. Computing speedup for large problems is difficult because they cannot fit into the memory of a single processor. For the bundle method, the parallel version performs the same calculations as the serial version, so we can estimate the serial execution time by timing the relevant portions of the parallel version. For the modified partition algorithm, however, the parallel version may not execute the same calculations as the serial version. In this case, we estimate the serial solution time by extrapolating data from problems that do fit on one processor. We feel that this is a reasonable estimate because of its near linear complexity.

Figure 8 shows the parallel execution time versus number of blocks for both algorithms. It shows that for each algorithm there is a range of problem sizes for which it outperforms the other. The curves are best understood by considering them in two parts: up to 1024 blocks, where one processor for each block is used; and greater than 1024 blocks, where the blocks are distributed amongst all 1024 processors. First, recall that the complexity of both the modified partition algorithm and the bundle method is approximately linear. This means that if the parallel algorithms had an efficiency of 1.0, then the solution time curve would be flat (for up to 1024 blocks). The flatness of the curve for the bundle method for up to 1024 processors reflects the fact that it is an efficient algorithm. The slope of the curve for the modified partition algorithm means that it is not as efficient as the bundle method. Beyond 1024 blocks, the curve for the bundle method rises and passes the curve for the partition algorithm. This happens because the efficiency of both algorithms increases as the number of blocks per processor increases, and because the serial partition algorithm outperforms the serial bundle method.

Figure 9 shows the scaled speedup for both algorithms. The scalability of the bundle method is very good. The speedup curve for it stays very near to the diagonal and does not show signs of leveling off. The curve for the modified partition algorithm is also good in that it does not show signs of leveling off, although it is not as close to the diagonal as the bundle method is.

The computational results presented permit a direct comparison of CPU time on the Cray X-MP and the elapsed time for the parallel solution on the nCUBE 2. This comparison for a range of problem sizes is shown in Table 1. It is seen that for the largest size problem the parallel solution is approximately 50 times faster. It should be noted that we did not attempt to vectorize the code on the Cray beyond what was done by the compiler. In any case, we do not consider the algorithm to be highly

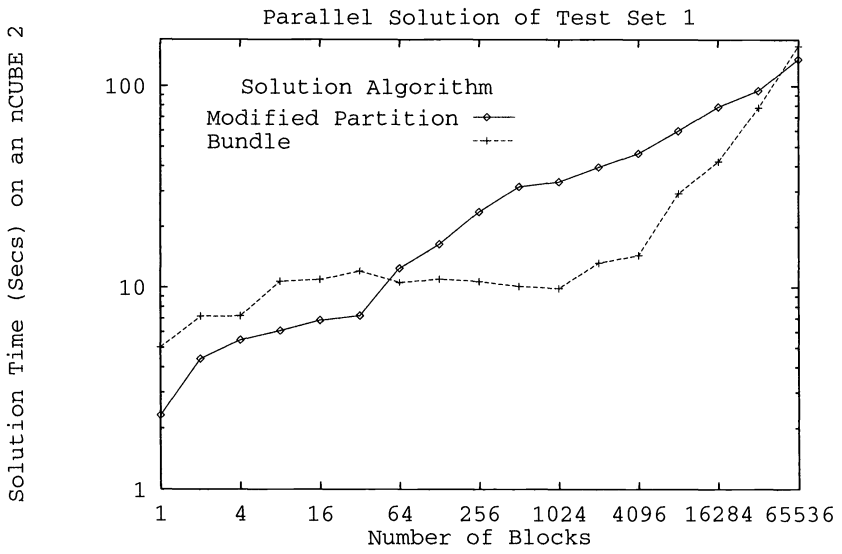


FIG. 8. *Performance of the parallel algorithms on problems with dense blocks.*

TABLE 1
Comparison of solution times on the Cray X-MP *and the* nCUBE 2.

Problem Size (Blocks)	Cray X-MP (CPU sec)	nCUBE 2 (elapsed sec)
64	5	12
512	54	32
65536	6900*	136

* – Estimated

vectorizable, since much of its time is spent in sparse matrix handling routines.

Appendix A. Test problem generation. Algorithm A.1 generates problems for testing the algorithms studied in this paper. These problems are of the form (1) with $f(y)$ linear and $c(y)$ quadratic. $f(y)$ is defined by $f(y) = b_0^T y$ and $c(y)$ is defined by $c(y) = d_0 - \frac{1}{2} D^T Y^2 e$, where $d_0 \in \Re^n$, $D \in \Re^{s \times n}$, $Y = \text{diag}(y)$, and e is a length s vector of all ones. Note that this choice for $c(y)$ gives the following quadratic function for each of its elements:

$$c_i(y) = d_{0,i} + \sum_{j=1}^{s} D_{j,i}\, y_j^2 \text{ for } i = 1, \ldots, n.$$

The generated problems also satisfy assumptions A1–A4. Note that making $f(y)$ linear can be done without loss of generality, as a nonlinear $f(y)$ can be shifted into the constraints with the addition of one variable and one constraint.

ALGORITHM A.1 (TEST PROBLEM GENERATOR)
Input – Parameters specifying the problem size (s, *mpblk*, *npblk*, and k); the density of the constraint matrix A (*den*); the solution (y^*, w^*); and a seed for the random number generator that selects the problem data. *mpblk* is the number of linear variables per block, and *npblk* is the number of constraints per block, so m = *mpblkk*, and

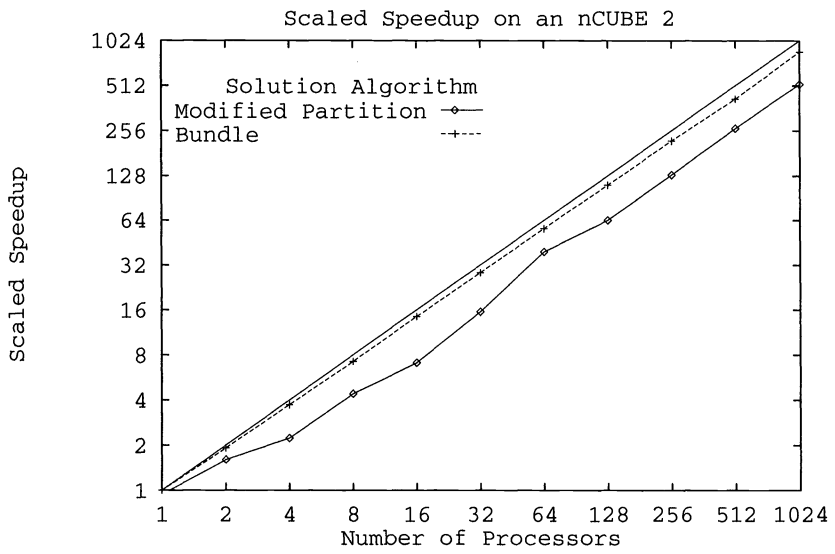


FIG. 9. *Scaled speedup of the modified partition algorithm and the bundle method on problems with dense blocks* (64 *blocks per processor*).

$n = npblkk$.

Output – The problem data, b_0, b, A, d_0, and D.

Begin

1. Randomly select members of A to be nonzero such that its density will be *den*. Choose random integers in $\{-9, \ldots, 9\}$ for the nonzero members of A. Test for singularity of A. If it is singular, repeat step 1. Otherwise, continue to step 2.
2. Select the first *mpblk* constraints of each block to be active at the solution.
3. Choose optimal Lagrange multipliers x of the problem. Set $x = 1/k$ for active constraints and $x = 0$ for inactive constraints.
4. Compute $b = Ax$.
5. Compute $c = A^T w^*$.
6. For constraints i that are inactive at the solution set $c_i = c_i + 1.0$.
7. Randomly select entries D to be nonnegative integers in $\{0, \ldots, 9\}$ and compute $d_0 = c + \frac{1}{2} D^T Y^2 e$.
8. Compute $b_0 = YDx$.
9. Compute the optimal function value $\Psi = b_0^T y^* + b^T w^*$.

End

The following three sets of test problems were generated.

1. Dense Jacobian, k varied.
   $s = 10$, $mpblk = 10$, $npblk = 20$, $den = 1.0$.
2. Sparse Jacobian, k varied.
   $s = 10$, $mpblk = 40$, $npblk = 80$, $den = 0.05$.
3. Dense Jacobian, s varied from 5 to 160.
   $k = 16, 64,$ and 256, $mpblk = 40$, $npblk = 80$, $den = 1.0$.

REFERENCES

[1] J. S. ARORA, *Computational design optimization: a review and future directions*, Structural Safety, 7 (1990), pp. 131–148.

[2] A. D. BELEGUNDU AND J. S. ARORA, *A recursive quadratic programming method with active set strategy for optimal design*, Internat. J. Numer. Methods Engrg., 20 (1984), pp. 803–816.

[3] Y. ERMOLIEV AND R. WETS, *Stochastic programming, an introduction*, in Numerical Techniques for Stochastic Optimization, Y. Ermoliev and R. Wets, eds., Springer-Verlag, Berlin, 1988, pp. 1–32.

[4] J. H. GLICK, *Scalable Parallel Solution of Very Large, Structured, Nonlinear Optimization Problems*, Ph.D. thesis, Computer Science Department, University of Minnesota, Minneapolis, MN, 1993.

[5] J. H. GLICK, R. S. MAIER, AND J. B. ROSEN, *Parallel solution of large-scale, block-diagonal concave maximization problems*, SIAM J. Optim., 1 (1991), pp. 501–514.

[6] K. C. KIWIEL, *Methods of Descent for Nondifferentiable Optimization*, Springer-Verlag, Berlin, 1985.

[7] ———, *Proximity control in bundle methods for convex nondifferentiable minimization*, Math. Programming, 46 (1990), pp. 105–122.

[8] L. S. LASDON, *Optimization Theory for Large Systems*, Macmillan, New York, 1970.

[9] C. LEMARECHAL, *Combining Kelley's and conjugate gradient methods*, Abstract, IX Internat. Symposium on Mathematical Programming, Budapest, 1976.

[10] ———, *Bundle methods in nonsmooth optimization*, in Nonsmooth Optimization, C. Lemarechal and R. Mifflin, eds., Pergamon Press, Oxford, 1978, pp. 79–102.

[11] ———, *On a bundle algorithm for nonsmooth optimization*, in Nonlinear Programming 4, O. L. Mangasarian, R. R. Meyer, and S. M. Robinson, eds., Academic Press, New York, 1981, pp. 245–282.

[12] ———, *Constructing bundle methods for convex optimization*, in FERMAT Days 85, J.-B. Hiriart-Urruty, ed., North-Holland, New York, 1986, pp. 201–240.

[13] C. LEMARECHAL AND J. J. STRODIOT, *Bundle methods, cutting-plane algorithms and sigma-Newton directions*, in Nondifferentiable optimization: motivations and applications, V. F. Demyanov and D. Pallaschke, eds., Springer-Verlag, Berlin, 1984, pp. 25–33.

[14] O. K. LIM AND J. S. ARORA, *An active set RQP algorithm for engineering design optimization*, Computer Meth. Appl. Mech. Engrg., 57 (1986), pp. 51–65.

[15] F. A. LOOTSMA AND K. M. RAGSDELL, *State-of-the-art in parallel nonlinear optimization*, Parallel Comput., 6 (1988), pp. 133–155.

[16] M. M. MAKELA, *On the methods of nonsmooth optimization*, in System Modelling and Optimization – Proceedings of the 14th IFIP Conference, H. J. Sebastian and K. Tammer, eds., Springer-Verlag, Berlin, 1990, pp. 177–186.

[17] R. MIFFLIN, *A modification and extension of Lemarechal's algorithm for nonsmooth minimization*, Math. Programming Study, 17 (1982), pp. 77–90.

[18] B. MURTAGH AND M. SAUNDERS, *MINOS 5.0 user's guide*, Tech. Report SOL 83-20, Department of Operations Research, Stanford University, Stanford, CA, 1983.

[19] B. N. PSHENICHNY AND Y. M. DANILIN, *Numerical Methods in Extremal Problems*, Mir, Moscow, 1978.

[20] J. B. ROSEN, *Convex partition programming*, in Recent Advances in Mathematical Programming, R. L. Graves and P. Wolfe, eds., McGraw Hill, New York, 1963, pp. 159–176.

[21] ———, *Primal partition programming for block diagonal matrices*, Numer. Math., 6 (1964), pp. 250–260.

[22] J. B. ROSEN AND J. C. ORNEA, *Solution of nonlinear programming problems by partitioning*, Management Sci., 10 (1963), pp. 160–173.

[23] H. SCHRAMM AND J. ZOWE, *A version of the bundle idea for minimizing a nonsmooth function: conceptual idea, convergence analysis, numerical results*, SIAM J. Optim., 2 (1992), pp. 121–152.

SIAM J. OPTIMIZATION
Vol. 4, No. 4, pp. 708–721, November 1994
© 1994 Society for Industrial and Applied Mathematics
002

EVALUATION OF LARGE-SCALE OPTIMIZATION PROBLEMS ON VECTOR AND PARALLEL ARCHITECTURES*

BRETT M. AVERICK† AND JORGE J. MORÉ‡

Abstract. The authors examine the importance of problem formulation for the solution of large-scale optimization problems on high-performance architectures. Limited memory variable metric methods are used to illustrate performance issues. It is shown that the performance of these algorithms is drastically affected by application implementation. Model applications are drawn from the MINPACK-2 test problem collection, with numerical results from a super-scalar architecture (IBM RS6000/370), a vector architecture (CRAY-2), and a massively parallel architecture (Intel DELTA).

Key words. optimization, large scale, limited memory, variable metric, performance evaluation, vector architecture, parallel architecture

AMS subject classifications. 65Y05, 65Y20, 65K05, 65K10, 90C06, 90C30

1. Introduction.

Our aim is to explore performance issues associated with the solution of large-scale optimization problems on high-performance architectures. The solution of these problems, where the number of variables ranges between 10^4 and 10^6, requires computer architectures with fast processors and large memories. We consider three architectures with these features: the IBM RS6000/370 workstation with 16 MW of memory is representative of a super-scalar architecture, the CRAY-2 with 512 MW of memory is representative of vector architectures, and the Intel DELTA (512 mesh-connected i860 processors) with a combined memory of 1024 MW is representative of massively parallel architectures.

We examine the importance of problem formulation for model applications drawn from the MINPACK-2 test problem collection. This collection is representative of large-scale optimization problems arising from applications in elasticity, combustion, lubrication, optimal design, superconductivity, and other fields of interest. Additional information on the MINPACK-2 problems can be found in Averick et al. [1].

Performance issues are illustrated with limited memory variable metric algorithms. These methods require minimal storage, lend themselves naturally to vector and parallel implementations and work well on a wide range of optimization problems. See, for example, the comparisons of Nash and Nocedal [17], and Zou et al. [20]. Our implementation of a limited memory variable metric (VMLM) algorithm, follows the approach of Liu and Nocedal [12]. In §3 we describe the key features of this code and its performance on the model applications of §2. The most noticeable aspect of these results is that on all the architectures under consideration, the cost of evaluating functions and gradients dominates the overall solution time. The obvious consequence of this observation is that improvements in performance can be obtained

* Received by the editors August 23, 1993; accepted for publication (in revised form) March 2, 1994.

† Cooper Neff Technologies, 3 Radnor Corporate Center, Suite 131, Radnor, Pennsylvania, 19087. The work of this author was supported by the Center for Research on Parallel Computation under National Science Foundation Cooperative Agreement CCR-8809615, and by Army Research Office Contract DAALO3-89-C-0038 with the University of Minnesota Army High Performance Computing Research Center.

‡ Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, Illinois 60439 (more@mcs.anl.gov). The work of this author was supported by Office of Scientific Computing, U.S. Department of Energy contract W-31-109-Eng-38.

only if the implementation of function and gradient evaluations takes advantage of the architecture.

Vectorization issues are treated in §§4 and 5. We show that vectorization of the function and gradient evaluation for our applications requires an appropriate partitioning of the element functions. These partitioning techniques are of interest because they are applicable to any partially separable function. In our applications they improve computing times by at least a factor of 10.

Sections 6 and 7 are concerned with distributed memory architectures. Section 6 shows that it is possible to develop algorithms for the function and gradient evaluation that are scalable in the sense that as the number of variables increases and the number of processors increases, the computing time remains reasonably constant. Section 7 shows that the implementation of the VMLM code is scalable in terms of the computing time per iteration. In §7 we also compare the performance of the VMLM code on all three architectures by determining when the DELTA is faster than the other two machines.

Performance evaluations of limited memory codes have also been performed by Nocedal [18] and Maier [13]. The issues brought out in these studies differ, in particular, because Nocedal deals only with shared memory architectures, while Maier's evaluation on the single instruction multiple data (SIMD) architecture of the CM-200 does not address scalability issues.

We end the paper by discussing possible extensions of this work. In particular, we point out that the performance issues that we have raised are also applicable to truncated Newton methods because the cost of evaluating functions and gradients for these methods is likely to dominate the computing time.

2. Large-scale optimization problems. The optimization problems that we consider arise from the need to minimize a function f of the form

$$f(v) = \int_{\mathcal{D}} \Phi(x, v, \nabla v)\, dx,$$

where $\mathcal{D}$ is some domain in $\mathbb{R}^2$ and Φ is defined by the application. In all cases f is well defined if $v : \mathcal{D} \mapsto \mathbb{R}^p$ belongs to $H^1(\mathcal{D})$, the Hilbert space of functions such that v and $\|\nabla v\|$ belong to $L^2(\mathcal{D})$. This is the proper setting for examining existence and uniqueness questions for the infinite-dimensional problem.

Finite element approximations to these problems are obtained by minimizing f over the space of piecewise linear functions v with values $v_{i,j}$ at $z_{i,j}$, $0 \le i \le n_y + 1$, $0 \le j \le n_x + 1$, where $z_{i,j} \in \mathbb{R}^2$ are the vertices of a triangulation of $\mathcal{D}$ with grid spacings h_x and h_y. The vertices $z_{i,j}$ are chosen to be a regular lattice so that there are n_x and n_y interior grid points in the coordinate directions, respectively.

Lower triangular elements T_L are defined by vertices $z_{i,j}, z_{i+1,j}, z_{i,j+1}$, while upper triangular elements T_U are defined by vertices $z_{i,j}, z_{i-1,j}, z_{i,j-1}$. The values $v_{i,j}$ are obtained by solving the minimization problem

$$\min \left\{ \sum \left(f_{i,j}^L(v) + f_{i,j}^U(v) \right) : v \in \mathbb{R}^n \right\},$$

where $f_{i,j}^L$ and $f_{i,j}^U$ are the finite element approximation to the integrals in the elements T_L and T_U, respectively.

Our problems are taken from the collection described in Averick et al. [1]. This report contains additional information on these problems; in particular, parameter values are chosen as in this report.

Steady state combustion (SSC). The study of the steady state in solid fuel ignition models leads to the optimization problem

$$\min\{f_\lambda(v) : v \in H_0^1(\mathcal{D})\},$$

where $f_\lambda : H_0^1(\mathcal{D}) \mapsto \mathbb{R}$ is the functional

$$f_\lambda(v) = \int_{\mathcal{D}} \left\{ \tfrac{1}{2}\|\nabla v(x)\|^2 - \lambda \exp[v(x)] \right\} dx$$

and $\lambda \geq 0$ is a parameter. The finite element approximation is defined by

$$f_{i,j}^L(v) = \frac{h_x h_y}{2} \left\{ \tfrac{1}{2} d_{i,j}^+(v) - \lambda \mu_{i,j}^+(v) \right\}, \qquad f_{i,j}^U(v) = \frac{h_x h_y}{2} \left\{ \tfrac{1}{2} d_{i,j}^-(v) - \lambda \mu_{i,j}^-(v) \right\},$$

where

$$d_{i,j}^\pm(v) = \left(\frac{v_{i\pm1,j} - v_{i,j}}{h_x} \right)^2 + \left(\frac{v_{i,j\pm1} - v_{i,j}}{h_y} \right)^2,$$

$$\mu_{i,j}^\pm = \tfrac{1}{3} \left\{ \exp(v_{i,j}) + \exp(v_{i\pm1,j}) + \exp(v_{i,j\pm1}) \right\}.$$

We use $\lambda = 2$ for our numerical experiments.

Minimal surface area (MSA). The determination of the surface with minimal area and given boundary values in a convex domain $\mathcal{D}$ is an optimization problem of the form

$$\min\{f(v) : v \in K\},$$

where $f : K \mapsto \mathbb{R}$ is the functional

$$f(v) = \int_{\mathcal{D}} \left(1 + \|\nabla v(x)\|^2 \right)^{\frac{1}{2}} dx,$$

and the set K is defined by

$$K = \left\{ v \in H^1(\mathcal{D}) : v(x) = v_D(x) \text{ for } x \in \partial\mathcal{D} \right\}$$

for some boundary data function $v_D : \partial\mathcal{D} \mapsto \mathbb{R}$. We consider the Enneper minimal surface defined on $\mathcal{D} = (-\frac{1}{2}, \frac{1}{2}) \times (-\frac{1}{2}, \frac{1}{2})$ by

$$v_D(\xi_1, \xi_2) = u^2 - v^2,$$

where u and v are the unique solutions to the equations

$$\xi_1 = u + uv^2 - \frac{1}{3}u^3, \qquad \xi_2 = -v - u^2 v + \frac{1}{3}v^3.$$

The finite element approximation is defined by

$$f_{i,j}^L(v) = \frac{h_x h_y}{2} \left\{ 1 + d_{i,j}^+(v) \right\}^{\frac{1}{2}}, \qquad f_{i,j}^U(v) = \frac{h_x h_y}{2} \left\{ 1 + d_{i,j}^-(v) \right\}^{\frac{1}{2}},$$

where

$$d_{i,j}^\pm(v) = \left(\frac{v_{i\pm1,j} - v_{i,j}}{h_x} \right)^2 + \left(\frac{v_{i,j\pm1} - v_{i,j}}{h_y} \right)^2.$$

Optimal design with composites (ODC). The optimal design problem requires determining the placement of two elastic materials in the cross section of a rod so as to maximize the resulting torsional rigidity. The problem is formulated in terms of a family of problems of the form

$$\min\{f_\lambda(v) : v \in H_0^1(\mathcal{D})\},$$

where $f_\lambda : H_0^1(\mathcal{D}) \mapsto \mathbb{R}$ is the functional

$$f_\lambda(v) = \int_{\mathcal{D}} \left\{ \psi_\lambda \left(\|\nabla v(x)\| \right) + v(x) \right\} dx,$$

and $\psi_\lambda : \mathbb{R} \mapsto \mathbb{R}$ is the piecewise quadratic

$$\psi_\lambda(t) = \begin{cases} \frac{1}{2}\mu_2 t^2, & 0 \le t \le t_1, \\ \mu_2 t_1 (t - \frac{1}{2} t_1), & t_1 \le t \le t_2, \\ \frac{1}{2}\mu_1 (t^2 - t_2^2) + \mu_2 t_1 (t_2 - \frac{1}{2} t_1), & t_2 \le t, \end{cases}$$

with the breakpoints t_1 and t_2 defined by

$$t_1 = \left(2\lambda \frac{\mu_1}{\mu_2} \right)^{\frac{1}{2}}, \qquad t_2 = \left(2\lambda \frac{\mu_2}{\mu_1} \right)^{\frac{1}{2}}.$$

We consider the problem of minimizing f_λ for a fixed value of $\lambda = 0.008$. In our numerical results we used $\mu_1 = 1$ and $\mu_2 = 2$ so that $t_1^2 = \lambda$, $t_2^2 = 2\lambda$. The finite element approximation is defined by

$$f_{i,j}^L(v) = \frac{h_x h_y}{2} \left[\psi_\lambda \left(e_{i,j}^+(v) \right) + \frac{1}{3} (v_{i,j} + v_{i+1,j} + v_{i,j+1}) \right],$$

$$f_{i,j}^U(v) = \frac{h_x h_y}{2} \left[\psi_\lambda \left(e_{i,j}^-(v) \right) + \frac{1}{3} (v_{i,j} + v_{i-1,j} + v_{i,j-1}) \right],$$

where

$$e_{i,j}^\pm(v) = \left\{ \left(\frac{v_{i\pm1,j} - v_{i,j}}{h_x} \right)^2 + \left(\frac{v_{i,j\pm1} - v_{i,j}}{h_y} \right)^2 \right\}^{1/2}.$$

3. Limited memory variable metric algorithms. Given an initial iterate x_0, variable metric methods generate iterates by

$$x_{k+1} = x_k - \alpha_k H_k \nabla f(x_k),$$

where $\alpha_k > 0$ is determined by a line search algorithm and where the matrix H_k is determined by an updating procedure. In a limited memory method the matrix H_k is not stored explicitly; instead, it is defined implicitly in terms of a fixed number of vectors.

In our implementation of the limited memory method, the search parameter α_k is determined by the `csrch` subroutine of Moré and Thuente [15]. The matrix H_{k+1} is defined, as in Liu and Nocedal [12], in terms of a scaling parameter σ_k and information gathered during the previous m iterations. Given

$$y_j = \nabla f(x_{j+1}) - \nabla f(x_j), \quad s_j = x_{j+1} - x_j, \quad \rho_j = y_j^T s_j, \qquad k - m + 1 \le j \le k,$$

the product $q = H_{k+1}w$ for any $w \in \mathbb{R}^n$ is computed with the pseudocode

$q \leftarrow w$
do $i = \min(k, m), \ldots, 1$
  $l \leftarrow i + \max(0, k - m)$
  $\beta_i \leftarrow (s_l^T q)/\rho_l$
  $q \leftarrow q - \beta_i y_l$
end do
$q \leftarrow \sigma q$
do $i = 1, \ldots, \min(k, m)$
  $l \leftarrow i + \max(0, k - m)$
  $q \leftarrow q + s_l \left(\beta_i - (y_l^T q)/\rho_l\right)$
end do

For most applications, the computational cost of limited memory algorithms is dominated by the cost of evaluating $f(x_k)$ and $\nabla f(x_k)$, where we measure cost in terms of floating-point operations (flops). In addition, our VMLM code uses $(8m + 1)n$ flops for the computation of $H_k \nabla f(x_k)$ and $11n$ flops for the remainder of the algorithm. Memory requirements are $(2m + 3)n$ words of storage.

Table 3.1 shows timing results (in seconds) for the solution of the test problems of §2 by VMLM with $m = 5$. The most striking feature of these results is the fraction of computing time required for the function evaluations. On one node of the DELTA the function evaluations require 75% of the solution time, on the CRAY-2 the function evaluations are responsible for over 90% of the overall solution time, and on the IBM RS6000 the functions require about 60% of the solution time. We obtain different ratios because the relative cost (in terms of flops) of evaluating the functions differs drastically from machine to machine. We address this point in §5. Another reason for the difference in ratios on the CRAY-2 is that the VMLM code vectorizes while the function evaluation code does not.

TABLE 3.1
Initial timings (seconds) for $n = 40,000$.

	DELTA (1 node)		CRAY-2		IBM RS6000/370	
Problem	function	algorithm	function	algorithm	function	algorithm
MSA	484	641	125	134	74	132
SSC	307	384	132	136	62	91
ODC	510	636	109	116	66	112

It is worthwhile noting that for our problems the number of VMLM iterations tends to grow like $n^{1/2}$. This can be seen clearly in Fig. 3.1, where we have plotted the number of iterations against $n^{1/2}$ for all three test problems. We also note that the linear dependence on $n^{1/2}$ for the number of iterations seems to be typical of two-dimensional grid problems. Another interesting observation on these problems is that in most cases the line search only requires one function evaluation, with the ratio of function evaluations to number of iterations always less than 1.05.

As a consequence of the above remarks, the time to solve grid problems with VMLM grows like $n^{3/2}$ on serial architectures. However, for parallel architectures our aim is to avoid the growth in time resulting from increasing problem size by increasing the number of processors. By developing a scalable (constant time per iteration) algorithm, the time to solve grid problems with VMLM grows like $n^{1/2}$.

In the remainder of the paper we look at several ways to improve the overall

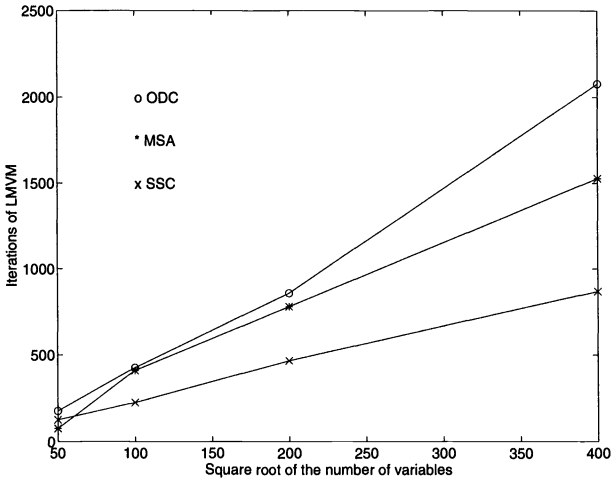


FIG. 3.1. *Number of iterations of* VMLM *on grid problems as a function of* $n^{1/2}$.

solution time of VMLM. Table 3.1 shows that to do this we must concentrate on improving the function and gradient evaluations.

4. Evaluating functions and gradients on vector architectures. The results of §3 clearly show that improving the performance of the VMLM code on the CRAY-2 requires vectorization of the function and gradient evaluations. In this section we show that this can be done by using partitioning techniques.

The standard way to evaluate the functions and gradients for the optimization problems discussed in §2 is to first sweep through the triangular elements with the pseudocode

```
do j = 1, ..., n_y
  do i = 1, ..., n_x
    Evaluate f^L_{i,j}(v) and ∇f^L_{i,j}(v)
    f(v) ← f(v) + f^L_{i,j}(v)
    ∇f(v) ← ∇f(v) + ∇f^L_{i,j}(v)
  end do
end do
```

followed by similar code for the triangular elements T_U. This code does not vectorize and thus performs poorly on the CRAY-2. The difficulty arises because functions $f^L_{i,j}$ and $f^L_{i+1,j}$ share the variables $v_{i+1,j}$ and thus both contribute to the same gradient element. More specifically, a vector dependency occurs because several elements try to write into the same component of the gradient.

The vector dependency can be eliminated by splitting the functions into groups such that no functions in a group depend on the same variable. This idea was suggested by Moré [14] for treating synchronization issues on shared memory architectures, but can easily be extended to vector architectures.

The *partition* problem of splitting the functions $f^L_{i,j}$ into groups so that functions in a group do not have variables in common is equivalent to partitioning the triangles T_L into groups so that triangles in a group do not intersect. For our test problems, if

$$S = \{(i,j) : 0 \le i \le n_x,\ 0 \le j \le n_y\},$$

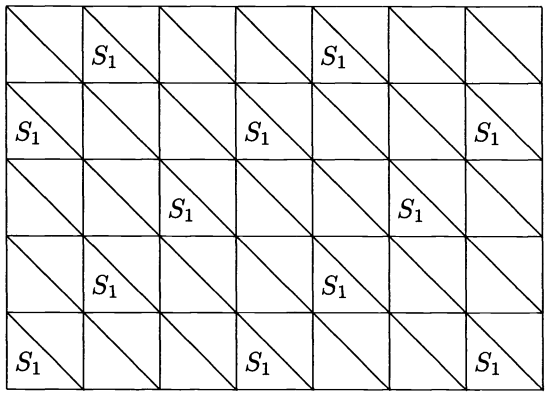


FIG. 4.1. *Triangular elements T_L in S_1.*

then the three sets

$$S_1 = \{(i,j) \in S : i = \text{mod}(j,3) + 3k,\ k = 0,1,\ldots\},$$
$$S_2 = \{(i,j) \in S : i = \text{mod}(j+1,3) + 3k,\ k = 0,1,\ldots\},$$
$$S_3 = \{(i,j) \in S : i = \text{mod}(j+2,3) + 3k,\ k = 0,1,\ldots\},$$

define a suitable partition. For example, the triangular elements T_L associated with S_1 are marked in Fig. 4.1.

The approach that we have outlined partitions the functions $f_{i,j}^L, f_{i,j}^U$ into six groups. This is the smallest number of groups as long as one of the variables is shared by six functions; in our case, all the variables appear in six functions. A standard red-black partitioning into four groups is therefore not possible. Of course, if we reformulate the optimization problem so that a function is associated with each square, then a standard red-black partitioning would be possible.

Table 4.1 shows the improvements that can be obtained on the CRAY-2 with the above partitioning. We used $n = 40,000$ variables on all problems and noted the time t_s to evaluate the standard f and ∇f and the time t_p to evaluate the partitioned f and ∇f.

TABLE 4.1
Function evaluation times (seconds) on the CRAY-2 for $n = 40,000$.

Problem	t_s	t_p
MSA	0.31	0.017
SSC	0.66	0.017
ODC	0.34	0.036

The increase in speed obtained by partitioning varies with each problem. For the MSA problem there is a speedup of 18, which is what we expect when comparing vectorized code with scalar code on the CRAY-2. The SSC problem shows a speedup of 39. This improvement is due not only to the partitioning, but also to the replacement of

$$\mu_{i,j}^{\pm} = \tfrac{1}{3}\{\exp(v_{i,j}) + \exp(v_{i\pm1,j}) + \exp(v_{i,j\pm1})\}$$

on all interior elements by

$$\mu_{i,j}^{\pm} = \exp(v_{i,j}).$$

Since each variable is a member of six elements, this modification does not change the SSC function (boundary values are handled separately). The ODC problem showed a speedup of 9. In this case vectorization of the ψ function computation required a modification to the original code that increased the flop count.

The partitioning approach can be extended to general partially separable functions, that is, functions of the form

$$f(v) = \sum_{k=1}^{n_p} f_k(v),$$

where each f_k depends on only a few of the variables v_i. For general partially separable functions, the partitioning problem can be formulated as a graph coloring problem. For details and references, see Coleman and Moré [5]. Other references in this area include Coleman, Garbow, and Moré [4] on software for the partition problem, Goldfarb and Toint [7] on the partition problem for matrices that arise from finite element approximations, Jones and Plassmann [10] on the use of partitioning techniques for the iterative solution of sparse linear systems, and Averick et al. [2] on computing large sparse Jacobian matrices using automatic differentiation.

5. Performance evaluation on vector architectures. We have shown that partitioning techniques can be used to improve the performance of the function and gradient evaluation algorithms on vector architectures. We now examine the performance of the VMLM code on the SSC problem on the basis of two measures: the time per iteration (t_{iter}) and the Mflop rate.

We already noted in §3 that the VMLM code requires $(8m + 12)n$ operations per iteration in addition to the flops needed to evaluate $f(x_k)$ and $\nabla f(x_k)$. For the SSC problem, the evaluation of $f(x_k)$ and $\nabla f(x_k)$ requires $38n$ operations and n evaluations of the exponential. The cost of evaluating the exponential function was determined experimentally by computing the times required to evaluate the SSC problem when a multiplication replaces the exponential function. We found that for our compiler options during these tests, an exponential evaluation costs (approximately) 20 multiplications on the IBM RS6000, 40 multiplications on the CRAY-2, and 130 multiplications on the DELTA.

An implementation of the exponential function is likely to require at least 30 flops. See, for example, the implementation described by Cody and Waite [3]. Thus, the number of operations on the CRAY-2 looks about right, while the number for the IBM probably reflects an extremely careful implementation. The DELTA is an experimental architecture, so the implementation of the intrinsic functions may not have received sufficient attention.

Table 5.1 presents the time per iteration (t_{iter}) and the Mflop rate on the IBM RS6000 and the CRAY-2. We have included the IBM RS6000 because these results are indicative of the performance that can be obtained with minimal effort on a high-performance workstation. Note, in particular, that the IBM results do not require partitioning of the function and gradient evaluation algorithms. We do not present results for the problem with $n = 2.56\ 10^6$ because it is too large for the 16 MW memory.

TABLE 5.1
Performance evaluation of the VMLM *code.*

	IBM RS6000/370		CRAY-2	
n	t_{iter}	Mflop	t_{iter}	Mflop
10,000	0.06	17.5	0.01	137
40,000	0.26	17.7	0.04	141
160,000	1.03	17.9	0.15	144
640,000	4.10	18.1	0.62	143
2,560,000	...	...	2.36	145

The performance results in Table 5.1 show that there is a slight increase in speed with increasing vector length and that the time per iteration scales linearly with n. The overall Mflop rates are quite satisfactory. For example, Dongarra and Van Der Vorst [6] mention that an unpreconditioned conjugate gradient iteration on a system of linear equations with $n = 10^6$ variables achieves 21.1 Mflops on an IBM RS6000/550 and 149 Mflops on the CRAY-2.

6. Evaluating functions and gradients on distributed memory architectures. Evaluation of the function and the gradient of a partially separable function on shared memory architectures has been discussed by Moré [14]. The issues are more complicated on distributed memory architectures; the discussion below follows the work of Jones and Plassmann [11].

The first issue that must be considered is that the work must be distributed among the processors so as to obtain good load balancing. In our test problems this amounts to distributing the evaluations of $f_{i,j}^L$ and $f_{i,j}^U$ evenly among the processors. Second, the variables need to be distributed among the processors so that the ratio of computation to communication remains high. That is, if a processor is responsible for computing $f_{i,j}^L$, it should own as many variables required by $f_{i,j}^L$ as possible.

Since the test problems are grid problems and the DELTA is a mesh-connected machine, the distribution of variables is fairly straightforward. We simply lay a grid of processors over the rectangular finite element mesh, as shown in Fig. 6.1, distributing the grid points as evenly as possible. In this figure each processor owns six variables.

Any element that has all of its vertices within one processor is an *interior* element; those elements with vertices in two or more processors are *shared* elements. Any element that has a vertex that lies on the boundary of the grid is a *boundary element.* It is possible to be a shared-boundary element, but these elements are to be considered shared.

Each processor is responsible for computing the function in all of its interior elements and in those shared elements for which the processor contains the right angle vertex. Processors that contain a vertex of a boundary element compute that boundary element's contributions to the function. Last, the function contributions of the bottom left and top right elements are computed by the appropriate processor. After each processor has computed its contribution to the function, the results are added to get the global function value.

Evaluating the gradient is less complicated. Each processor is responsible for computing each of the gradient components corresponding to the variables that lie within the processor. This means that a processor must compute the gradient con-

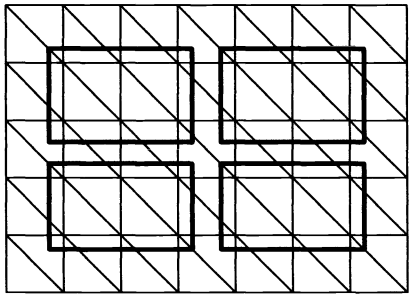


FIG. 6.1. *Partitioning variables into processors.*

tributions of some shared elements even though the processor does not compute the corresponding function.

Figure 6.2 shows a typical processor with six variables. This processor is responsible for computing the function for all the elements shown. The gridpoints labeled **G**, frequently called *ghostpoints*, are needed to compute the contribution to the function and the component of the gradient for the variables owned by the processor. The ghostpoints must be obtained from the processors directly above, directly below, immediately to the right, and immediately to the left. These data must be communicated to the processor before it can compute the functions or gradient components that depend on the ghostpoints. Communication is handled by the *Chameleon* parallel programming tools developed by Gropp and Smith [8]. These tools can be used to develop code that is portable to a variety of computing environments.

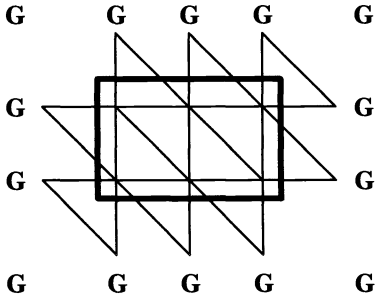


FIG. 6.2. *Processor with ghostpoints.*

Let m_x and m_y be the number of grid points in the respective coordinate directions owned by the processors so that $\texttt{nvp} = m_x m_y$ is the number of variables stored in each processor. We store the vector of variables x as an $(m_y+2)\times(m_x+2)$ array and the gradient as an $m_y \times m_x$ array. The extra rows and columns in x allow space for the ghostpoints. This strategy worked well for our problems but may be impractical in terms of storage for three-dimensional problems or for elements of high order.

For a fixed problem size, the computing time for evaluating the SSC function and gradient depends on the number of variables **nvp** stored in each processor. Figure 6.3 shows the time for **nvp** set to $5,000, 10,000$, and $20,000$. The computing times decrease as **nvp** decreases since more processors are being used; however, it is not always possible or convenient to use all the DELTA processors, so the results for $\texttt{nvp} = 20,000$ are of interest although they do not lead to the fastest computing times.

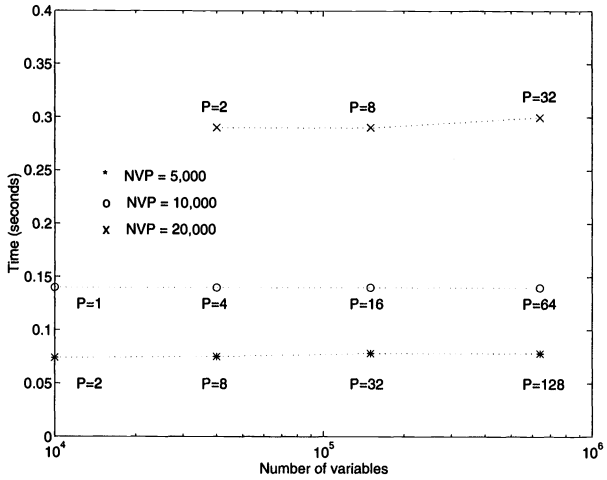


FIG. 6.3. *Evaluation of the* SSC *function on the* DELTA.

Figure 6.3 shows that the function and gradient evaluation algorithm is scalable in the sense that as the number of variables increases and the number of processors increases, the computing time remains reasonably constant. Furthermore, the uniformity with which the plots are spaced indicates that the communication time needed to do the evaluation is almost negligible, even for small values of nvp.

7. Performance evaluation on parallel architectures. We have noted that the evaluation of the function and gradient on a distributed memory architecture requires that the vector x of variables and the gradient vector g be distributed among the processors. Implementation of the VMLM code also requires that the $2m$ vectors y_i and s_i be distributed among the processors. All other information is local to each processor. With this strategy, processors are required to store vectors of length proportional to nvp.

Global communication is required to compute the quantities $g^T s$ and $y^T s$. The line search needs $g^T s$, and $y^T s$ is used to determine the scaling parameter. Global communication is also needed to compute $s_i^T q$ and $y_i^T q$ during the computation of the direction d (see the algorithm in §3). Thus, each iteration of the LMVM code requires $2(m+1)$ global dot products. These computations are performed with the *Chameleon* package of Gropp and Smith [8].

The line search routine requires very few flops except for function and gradient evaluations, which we have already shown to be amenable to parallel implementation. As a result each processor performs exactly the same line search in our VMLM implementation, and parallelism is obtained through the parallelism in the function and gradient evaluations.

Table 7.1 presents the performance data for the VMLM code with nvp = 5000. The time per iteration (t_{iter}) is virtually constant, indicating scalability of performance. There is an increase in the time per iteration for the largest problem; it is generally believed that at the time these results were obtained, the DELTA had a faulty communication channel. Results were also obtained for nvp = 10,000 and nvp = 20,000. The results scale almost linearly with those shown in Table 7.1. For example, the average time per iteration increases from 0.14 for nvp = 5,000 to 0.26

TABLE 7.1
Performance evaluation of VMLM *on the* DELTA (**nvp** = 5,000).

n	p	t_{iter}	Mflop/p	Mflops
10,000	2	0.12	9.3	18
40,000	8	0.13	8.8	71
160,000	32	0.13	8.3	267
640,000	128	0.14	8.1	1044
2,560,000	512	0.18	6.3	3233

for **nvp** = 10,000, and to 0.50 for **nvp** = 20,000.

Figure 7.1 compares the results in Table 7.1 with those presented in Table 5.1 for the IBM RS6000 and the CRAY-2. We show DELTA results for **nvp** = 5,000 and **nvp** = 10,000. Initially both sequential machines are faster than the DELTA because of processor speed, but this situation is reversed as the problem size increases and the DELTA adds processors. First consider the case when **nvp** = 5,000. The DELTA (4 processors) becomes faster than the IBM RS6000 at $n = 20,000$, while the DELTA (28 processors) becomes faster than the CRAY-2 at $n = 140,000$. The crossing point for other values of **nvp** can be determined from this data because computing times scale linearly with n and **nvp**. For example, with **nvp** = 10,000, the DELTA (28 processors) becomes faster than the CRAY-2 at $n = 280,000$.

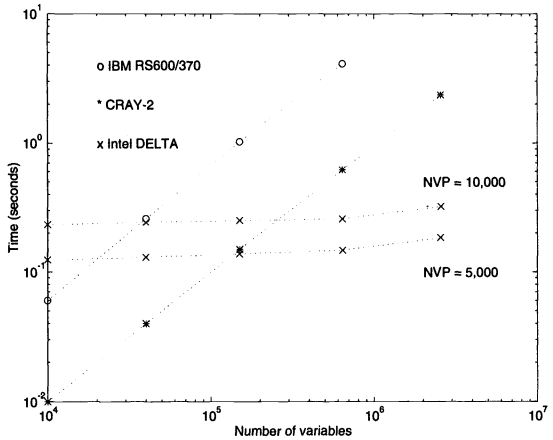


FIG. 7.1. *Time (seconds) per iteration of* VMLM *as a function of* n.

8. Concluding remarks. The performance issues that we have raised apply to truncated Newton methods because, for these methods, the cost of evaluating the function and gradient is likely to dominate the computing time. Nash and Nodedal [17] mention that a typical cost of the TN code of Nash [16] is $325n$ flops per iteration. This cost is certainly higher than for VMLM methods, but we still expect it to be a small fraction of the total computing cost for many applications. The cost of the truncated Newton method of Schlick and Fogelson [19] depends on the preconditioner, so the situation is less clear with this code.

TABLE 8.1
Evaluation of $\nabla^2 f(x)v$ *(seconds) for the* SSC *problem on the* CRAY-2.

n	t_s	t_p
10,000	.18	.004
40,000	.70	.015
160,000	2.79	.061
640,000	11.20	.244

Although we have concentrated on issues involving the evaluation of the function and gradient, many of these issues also apply to the evaluation of the Hessian-vector product $\nabla^2 f(x)v$. This product is often required by, for example, truncated Newton methods. In most codes, this product is approximated by differences of gradient values, but it is certainly possible and desirable to use the actual Hessian-vector product, since this eliminates the dependence of the code on a difference parameter.

The MINPACK-2 test problems include code for the evaluation of $\nabla^2 f(x)v$. These codes raise the same issues as the gradient evaluation codes. In particular, on vector architectures these codes have the same vector dependencies as the gradient evaluation codes. These dependencies can be eliminated by the same partitioning techniques discussed in §4. Table 8.1 shows the improvements that can be obtained with these partitioning techniques. We used the SSC problem with $n = 40,000$ variables and noted the time t_s to evaluate the standard $\nabla^2 f(x)v$ and the time t_p to evaluate the partitioned $\nabla^2 f(x)v$.

Note that the computing time scales linearly with n. Moreover, note that Table 4.1 shows that the cost of evaluating the function and the gradient is about the same as the cost of evaluating the Hessian-vector product. This is typical for our applications and should hold in all cases. For a discussion of this point, in connection with automatic differentiation, see Iri [9].

As a final remark we note that the evaluation of Hessian-vector products on distributed memory architectures can also be done with the techniques discussed in §6, and that results were obtained for the SSC problem that were similar to those shown in Fig. 6.3.

Acknowledgments. We thank Bill Gropp, Mark Jones, Barry Smith, and Paul Plassmann for sharing their parallel programming expertise, and Peter Tang for his observations on algorithms for evaluating intrinsic functions. Michelle Hribar deserves a special thanks for providing much of the initial programming for this project.

REFERENCES

[1] B. M. AVERICK, R. G. CARTER, J. J. MORÉ, AND G.-L. XUE, *The MINPACK-2 test problem collection*, Preprint MCS-P153-0692, Argonne National Laboratory, Argonne, IL, 1992.

[2] B. M. AVERICK, J. J. MORÉ, C. H. BISCHOF, A. CARLE, AND A. GRIEWANK, *Computing large sparse Jacobian matrices using automatic differentiation*, Preprint MCS-P348-0193, Argonne National Laboratory, Argonne, IL, 1993.

[3] W. J. CODY AND W. WAITE, *Software Manual for the Elementary Functions*, Prentice-Hall, Englewood Cliffs, NJ, 1980.

[4] T. F. COLEMAN, B. S. GARBOW, AND J. J. MORÉ, *Software for estimating sparse Jacobian matrices*, ACM Trans. Math. Software, 10 (1984), pp. 329–345.

[5] T. F. COLEMAN AND J. J. MORÉ, *Estimation of sparse Jacobian matrices and graph coloring problems*, SIAM J. Numer. Anal., 20 (1983), pp. 187–209.

[6] J. J. DONGARRA AND H. A. VAN DER VORST, *Performance of various computers using standard sparse linear equation solving techniques*, in Computer Benchmarks, J. J. Dongarra and H. A. Van Der Vorst, eds., North-Holland, Amsterdam, 1994, pp. 177–191.

[7] D. GOLDFARB AND P. L. TOINT, *Optimal estimation of Jacobian and Hessian matrices that arise in finite difference calculations*, Math. Comp., 43 (1984), pp. 69–88.

[8] W. GROPP AND B. SMITH, *Users manual for the Chameleon parallel programming tools*, Report ANL-93/23, Argonne National Laboratory, Argonne, IL, 1993.

[9] M. IRI, *History of automatic differentiation and rounding error estimation*, in Automatic Differentiation of Algorithms, A. Griewank and G. F. Corliss, eds., Society for Industrial and Applied Mathematics, 1992, Philadelphia, pp. 3–16.

[10] M. T. JONES AND P. E. PLASSMANN, *Scalable iterative solution of sparse linear systems*, Preprint MCS-P277-1191, Argonne National Laboratory, Argonne, IL, 1991.

[11] ———, *Computation of equilibrium vortex structures for type-*II *superconductors*, Internat. J. Supercomputing Appl., 7 (1993), to appear.

[12] D. C. LIU AND J. NOCEDAL, *On the limited memory* BFGS *method for large scale optimization*, Math. Programming, 45 (1989), pp. 503–528.

[13] R. S. MAIER, *Large-scale minimization on the* CM-200, Optim. Methods Software, 1 (1992), pp. 55–69.

[14] J. J. MORÉ, *On the performance of algorithms for large-scale bound constrained problems*, in Large-Scale Numerical Optimization, T. F. Coleman and Y. Li, eds., Society for Industrial and Applied Mathematics, Philadelphia, 1991, pp. 31–45.

[15] J. J. MORÉ AND D. J. THUENTE, *Line search algorithms with guaranteed sufficient decrease*, Preprint MCS-P330-1092, Argonne National Laboratory, Argonne, IL, 1992.

[16] S. G. NASH, *Preconditioning of truncated Newton methods*, SIAM J. Sci. Statist. Comput., 6 (1985), pp. 599–616.

[17] S. G. NASH AND J. NOCEDAL, *A numerical study of the limited memory* BFGS *method and the truncated Newton method for large scale optimization*, SIAM J. Optim., 1 (1991), pp. 358–372.

[18] J. NOCEDAL, *The performance of several algorithms for large-scale unconstrained optimization*, in Large-Scale Numerical Optimization, T. F. Coleman and Y. Li, eds., Society for Industrial and Applied Mathematics, 1991, Philadelphia, pp. 138–151.

[19] T. SCHLICK AND A. FOGELSON, *TNPACK – A truncated Newton minimization package for large-scale problems:* I. *Algorithms and usage*, ACM Trans. Math. Software, 18 (1992), pp. 46–70.

[20] X. ZOU, I. M. NAVON, M. BERGER, K. H. PHUA, T. SCHLICK, AND F. X. LE DIMET, *Numerical experience with limited-memory quasi-Newton and truncated Newton methods*, SIAM J. Optim., 3 (1993), pp. 582–608.

SIAM J. OPTIMIZATION
Vol. 4, No. 4, pp. 722–734, November 1994
© 1994 Society for Industrial and Applied Mathematics
003

SERIAL AND PARALLEL MULTICATEGORY DISCRIMINATION*

KRISTIN P. BENNETT† AND O. L. MANGASARIAN†

Abstract. A parallel algorithm is proposed for a fundamental problem of machine learning, that of multicategory discrimination. The algorithm is based on minimizing an error function associated with a set of highly structured linear inequalities. These inequalities characterize piecewise-linear separation of k sets by the maximum of k affine functions. The error function has a Lipschitz continuous gradient that allows the use of fast serial and parallel unconstrained minimization algorithms. A serial quasi-Newton algorithm is considerably faster than previous linear programming (LP) formulations. A parallel gradient distribution algorithm is used to parallelize the error-minimization problem. Preliminary computational results are given for both a DECstation 5000/125 and a Thinking Machines Corporation CM-5 multiprocessor.

Key words. multicategory classification, parallel optimization, machine learning

AMS subject classifications. 90B50, 90C20

1. Introduction.

We consider a fundamental problem of machine learning and pattern recognition, that of discriminating between k sets. Given k disjoint sets, $\mathcal{A}^i$, $i = 1, \ldots, k$, in the n-dimensional real space R^n, the problem is to construct a function that discriminates between these k sets. The function can then be used to classify future points that belong to one of the sets. We propose a piecewise-linear convex function that is the maximum of k linear (affine) functions. This function has proven to be very useful in decision tree learning methods [6]. In [4], an LP approach was proposed for constructing the function by minimizing the average classification error. In this work we formulate a 2-norm approach that involves the minimization of an unconstrained piecewise-quadratic convex function with a Lipschitz-continuous gradient. The two principal advantages of the new formulation over the LP approach are that (i) the serial version of the new approach is much faster than the LP formulation, and (ii) the new approach is much easier to parallelize via an iterative parallel gradient distribution algorithm [16]. Nilsson [21], Duda and Fossum [7], Duda and Hart [8], and Fukunaga [12] considered iterative methods that are extensions of the perceptron algorithm or the Motzkin–Schoenberg algorithm [18] for determining a piecewise-linear separator provided that one exists. Unlike our proposed approach, convergence of these iterative methods is not known if a separating piecewise-linear surface does not exist [12, p. 374].

We give now an outline of the paper. In §2 we review the LP formulation (5) for piecewise-linear separation using the 1-norm error formulation and then give the 2-norm formulation (6). We establish a new simple condition (9) for the occurrence of the null solution for the 2-norm problem (6), which turns out to be equivalent to that for the 1-norm formulation (5) [4]. In §3, we discuss both serial and parallel algorithms for solving the optimization problem and compare this formulation with previous approaches. The pertinent theorem (Theorem 3.1) of the parallel gradient distribution method [16] is given and an algorithm based on it is described (Theorem 3.2). Details of the algorithms and implementation are given in §3. Section 4 gives

* Received by the editors July 20, 1993; accepted for publication (in revised form) January 28, 1994.

† Computer Sciences Department, University of Wisconsin, 1210 West Dayton Street, Madison, Wisconsin 53706 (bennett@cs.wisc.edu, olvi@cs.wisc.edu). This material is based on research supported by Air Force Office of Scientific Research grant AFOSR-89-0410 and National Science Foundation grants CCR-9101801 and CDA-9024618.

a serial computational comparison of the LP 1-norm approach and the new 2-norm approach, as well as the results of the proposed parallel algorithm implemented serially and in parallel on the Thinking Machines CM-5 parallel processor.

Our notation is described now. For a vector x in the h-dimensional real space R^h, x_+ denotes the vector in R^h with components $(x_+)_i := \max\{x_i, 0\}$, $i = 1, \ldots, h$. The notation $A \in R^{m \times h}$ signifies a real $m \times h$ matrix. For such a matrix, A^T denotes the transpose while A_i denotes the ith row. A vector of ones in the real space R^{m^i}, $m = \sum_{i=1}^{k} m^i$, will be denoted by e^i. The 2-norm is denoted by $\|.\|_2$, while the 1-norm is denoted by $\|.\|_1$. The sequence $\{x_i\}$, $i = 0, 1, \ldots$, represents iterates in the h-dimensional real space R^h generated by some algorithm. For $\ell = 1, \ldots, k$, $x_i^\ell \in R^{h^\ell}$ represents an h^ℓ-dimensional subset of components of x_i, where $\sum_{\ell=1}^{k} h^\ell = h$. The complement of ℓ in $\{1, \ldots, k\}$ will be denoted by $\bar{\ell}$ and we write $x_i = (x_i^\ell, x_i^{\bar{\ell}})$, $\ell = 1, \ldots, k$. For a differentiable function $f: R^h \to R$, ∇f denotes the h-dimensional vector of partial derivatives with respect to x, and $\nabla_\ell f$ denotes the h^ℓ-dimensional vector of partial derivatives with respect to $x^\ell \in R^{h^\ell}$, $\ell = 1, \ldots, k$. For k points y_j, $j = 1, \ldots, k$ in R^h, the point $\sum_{j=1}^{k} \lambda_j y_j$, such that $\lambda_j > 0$ and $\sum_{j=1}^{k} \lambda_j = 1$ is said to be a strong convex combination of the points y_j, $j = 1, \ldots, k$. If f has continuous first partial derivatives on R^h, we write $f \in C^1(R^h)$.

2. Multicategory separation by a piecewise-linear surface. We begin by defining the concept of piecewise-linear separation of k sets in R^n [21], [4] and formulating the problem of minimizing the 1-norm error as a linear program.

DEFINITION 2.1 (Piecewise-linear separability). *The k sets $\mathcal{A}^i$, $i = 1, \ldots, k$, each consisting of m^i, $i = 1, \ldots, k$, points in R^n and represented by the $m^i \times n$ matrices, A^i, $i = 1, \ldots, k$, are piecewise-linear separable if there exist $w^i \in R^n$, $\gamma^i \in R$, $i = 1, \ldots, k$ such that*

$$A^i w^i - e^i \gamma^i \geq A^i w^j - e^i \gamma^j + e^i, \quad i, j = 1, \ldots, k, \ i \neq j. \tag{1}$$

Equivalently, there exists a piecewise-linear convex function determined by (w^i, γ^i), $i = 1, \ldots, k$, such that

$$p(x) = \max_{1 \leq \ell \leq k} x w^\ell - \gamma^\ell, \tag{2}$$

and

$$\left\langle \begin{array}{l} p(x) = x w^i - \gamma^i \\ p(x) > x w^j - \gamma^j \end{array} \right\rangle \begin{array}{l} \text{for } x \in \mathcal{A}^i, \ i = 1, \ldots, k, \\ \text{for } j \neq i. \end{array} \tag{3}$$

In [4], it was shown that the inequalities of the piecewise-linear separator are satisfied if and only if the minimum of the 1-norm of the average violations of the inequalities (1) is zero, namely,

$$0 = \min_{w^i, \gamma^i} \sum_{i=1}^{k} \sum_{\substack{j=1 \\ j \neq i}}^{k} \frac{e^i}{m^i} \left(-A^i(w^i - w^j) + e^i(\gamma^i - \gamma^j) + e^i\right)_+. \tag{4}$$

This minimization problem can be written as the following linear program (LP):

$$\min_{w^i, \gamma^i, y^{ij}} \left\{ \sum_{i=1}^{k} \sum_{\substack{j=1 \\ j \neq i}}^{k} \frac{e^i y^{ij}}{m^i} \;\middle|\; \begin{array}{c} y^{ij} \geq -A^i(w^i - w^j) + e^i(\gamma^i - \gamma^j) + e^i, \ y^{ij} \geq 0, \\ i \neq j, \ i, j = 1, \ldots, k \end{array} \right\}. \tag{5}$$

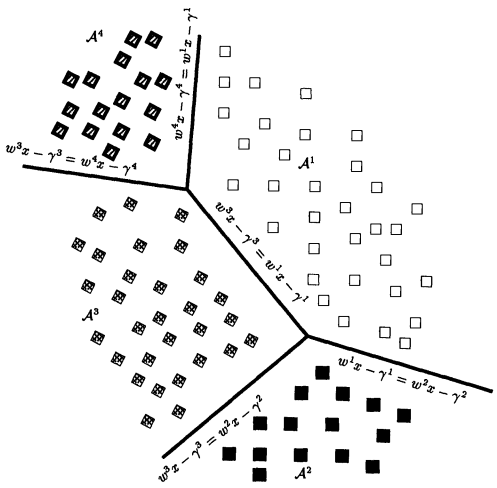


FIG. 1. *Piecewise-linear separator of four classes in* R^2.

Figure 1 depicts the piecewise-linear classifier found by the LP (5) for a typical piecewise-linear separable case with $k = 4$ and $n = 2$.

As shown in [4], the linear program (5) is quite effective on real-world problems. In practice such problems are rarely piecewise-linear separable and thus a multivariate decision tree must be used. A multivariate decision tree works by applying the linear program or another algorithm to a k-class classification problem. The resulting piecewise-linear surface divides the space into k regions. If each of these k regions contains mostly points of one class, then we are done. If any region contains an unacceptable mixture of points, then the linear program (5) or the other algorithm is used again to divide that region into k or fewer regions. The resulting discriminant function can be thought of as a decision tree. Figure 2 illustrates a decision surface found by a multivariate decision tree algorithm and Figure 3 depicts the corresponding decision tree. Brodley and Utgoff [6], [26] found that multivariate decision trees using piecewise-linear separators generalized better than the popular univariate decision tree algorithms CART and ID3 [5], [22]. As in other decision tree algorithms, pruning and feature selection need to be incorporated into the decision tree algorithm to prevent over-fitting [23], [6], [17]. This paper concentrates on the computational aspects of the problem of constructing the piecewise-linear surface at each node of the decision tree.

Although the linear program (5) is effective for use in multivariate decision tree algorithms, it can be very slow because the LP problem size can get quite large. Specifically for a problem with m points in R^n that belong to k classes, there are $m \times (k-1)$ constraints and $m \times (k-1) + k \times (n+1)$ variables (not counting slacks). Since many such linear programs may be needed to find a single decision tree, a fast method is desired. Ideally, an iterative method is also desirable in case new points are added and the tree needs to be adjusted [25]. Previous iterative approaches based on extensions to the perceptron algorithm [21], [7], [8] do not have stable performance for the inseparable case and, as a result, heuristic methods [13], [6] have been developed to get around this deficiency. Ideally, we would like to have a fast parallelizable algorithm that can be shown to converge for both the separable and the more common inseparable problems. By starting from Definition 2.1 and reformulating the problem,

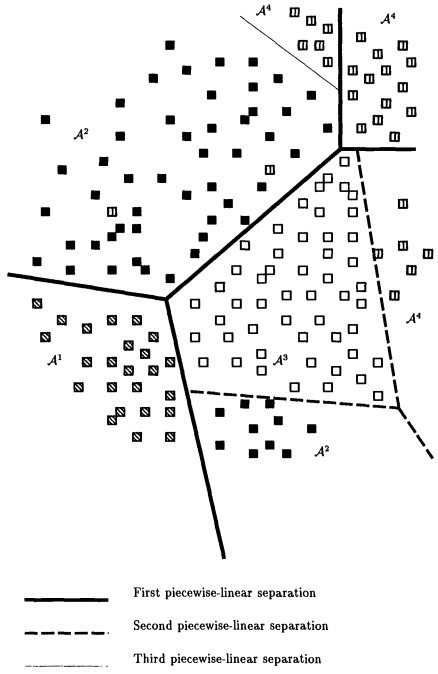


FIG. 2. *Geometric depiction of decision tree consisting of three piecewise-linear separators distinguishing four classes in* R^2.

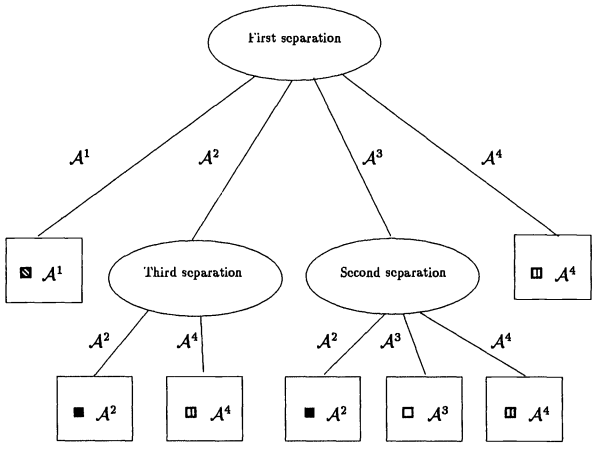


FIG. 3. *Decision tree representing the three piecewise-linear separators depicted in Fig.* 2.

we can accomplish this.

Consider the 2-norm formulation of minimizing of the average violation:

$$\min_{w^i,\gamma^i} f(w,\gamma) = \frac{1}{2}\sum_{i=1}^{k}\sum_{\substack{j=1\\ j\neq i}}^{k}\frac{1}{m^i}\left\|\left(-A^i(w^i-w^j)+e^i(\gamma^i-\gamma^j)+e^i\right)_+\right\|_2^2. \tag{6}$$

Squaring the plus function results in a piecewise-quadratic function that is differentiable. We refer to (6) as the piecewise-quadratic minimization (PQM) problem. The

first partial derivative of the function (below) can be shown to be Lipschitz continuous. In particular, we have

$$\begin{aligned}\nabla_{w^\ell} f(w,\gamma) \;=\; & \sum_{\substack{j=1\\ j\neq \ell}}^{k} \frac{-1}{m^\ell}\, A^{\ell T}\big(-A^\ell(w^\ell - w^j) + e^\ell(\gamma^\ell - \gamma^j) + e^\ell\big)_+ \\ & + \sum_{\substack{i=1\\ i\neq l}}^{k} \frac{1}{m^i}\, A^{i T}\big(-A^i(w^i - w^\ell) + e^i(\gamma^i - \gamma^\ell) + e^i\big)_+,\end{aligned} \tag{7}$$

$$\begin{aligned}\nabla_{\gamma^\ell} f(w,\gamma) \;=\; & \sum_{\substack{j=1\\ j\neq \ell}}^{k} \frac{1}{m^\ell}\, e^\ell\big(-A^\ell(w^\ell - w^j) + e^\ell(\gamma^\ell - \gamma^j) + e^\ell\big)_+ \\ & + \sum_{\substack{i=1\\ i\neq \ell}}^{k} \frac{-1}{m^i} e^i\big(-A^i(w^i - w^\ell) + e^i(\gamma^i - \gamma^\ell) + e^i\big)_+.\end{aligned} \tag{8}$$

As in the 1-norm formulation, the inequalities (1) for piecewise-linear separation hold if and only if the minimum of (6) is zero. Consequently, the following theorem holds.

THEOREM 2.1 (Multicategory separation via PQM). *The sets* $\mathcal{A}^i$, $i = 1,\dots,k$, *represented by the* $m^i \times n$ *matrices* A^i, $i = 1,\dots,k$, *are piecewise-linear separable if and only if the solvable piecewise quadratic minimization* (6) *has a zero minimum, in which case any solution* $(w^i,\ \gamma^i)$, $i = 1,\dots,k$, *provides a piecewise-linear separation as characterized in Definition* 2.1.

As was the case for linear and piecewise-linear separation of two sets by linear programming [2], [3], it is important to determine when the useless null solution occurs for sets $\mathcal{A}^i$, $i = 1,\dots,k$ that are not piecewise-linear separable. Note that the piecewise-linear separation (1) is achieved by a special pairwise linear separation between the sets $\mathcal{A}^i$, $i = 1,\dots,k$, that is determined by $(w^i - w^j,\ \gamma^i - \gamma^i) \in R^n \times R^1$, $i \neq j$, $i,\ j = 1,\dots,k,$. It is therefore the nonzeroness of $w^i - w^j$, $i \neq j$, $i,\ j = 1,\dots,k$ that matters. Nonzeroness of $w^i - w^j$, $i \neq j$, $i,\ j = 1,\dots,k$, is an important issue when one is trying to generate an approximate piecewise-linear separation (i.e., allow some errors in the separation) for sets that are not piecewise-linear separable. Zero $w^i - w^j$, $i \neq j$, $i,\ j = 1,\dots,k$ yields no information and hence no approximate separation for this case is obtained.

We now give a result that provides a necessary and sufficient condition for the occurrence of the null solution: $w^i - w^j = 0$, $i \neq j$, $i,\ j = 1,\dots,k$.

THEOREM 2.2 (Null solution occurrence). *The piecewise* PQM (5) *has the null solution,* $w^i - w^j = 0$, $i \neq j$, $i,\ j = 1,\dots,k$ *if and only if all class means are equal, that is*

$$\frac{e^i A^i}{m^i} = \frac{e^j A^j}{m^j}, \quad i = 1,\dots,k, \quad j = 1,\dots,k. \tag{9}$$

Proof. The vectors $w^i - w^j = 0$, $i \neq j$, $i,\ j = 1,\dots,k$ constitute an optimal solution of problem (6) if and only if $\gamma^i = \gamma^j = 0$, $i \neq j$, $i, j = 1,\dots,k$. For these values of $w^i, \gamma^i, i = 1,\dots,k$,

$$\nabla_{w^\ell} f(w,\gamma) \;=\; 0, \qquad \nabla_{\gamma^\ell} f(w,\gamma) \;=\; 0,\ \ell = 1,\dots,k. \tag{10}$$

Evaluating the partial gradients at such an optimal point gives

$$\nabla_{w^\ell} f(w,\gamma) = \sum_{\substack{j=1 \\ j\neq\ell}}^{k} -\frac{1}{m^\ell} A^{\ell T} e^\ell + \sum_{\substack{i=1 \\ i\neq\ell}}^{k} \frac{1}{m^i} A^{iT} e^i$$

(11)

$$= (1-k)\frac{A^{\ell T} e^\ell}{m^\ell} + \sum_{\substack{i=1 \\ i\neq\ell}}^{k} \frac{1}{m^i} A^{iT} e^i = 0,$$

$$\text{(12)} \quad \nabla_{\gamma^\ell} f(w,\gamma) = \sum_{\substack{j=1 \\ j\neq\ell}}^{k} \frac{1}{m^\ell} e^\ell e^\ell - \sum_{\substack{i=1 \\ i\neq\ell}}^{k} \frac{1}{m^i} e^i e^i = (k-1) - (k-1) = 0$$

Equation (12) is automatically satisfied. Obviously, (9) implies (11). To show the converse, suppose that condition (9) does not hold. Without loss of generality let

$$\frac{e^1 A^1}{m^1} > \frac{e^2 A^2}{m^2}.$$

If condition (11) holds, then we have the contradiction

(13)

$$(k-1)\frac{e^1 A^1}{m^1} > (k-1)\frac{e^2 A^2}{m^2} = \frac{e^1 A^1}{m^1} + \sum_{j=3}^{k} \frac{e^j A^j}{m^j} > \frac{e^2 A^2}{m^2} + \sum_{j=3}^{k} \frac{e^j A^j}{m^j} = (k-1)\frac{e^1 A^1}{m^1}.$$

Hence (11) does not hold and the proof is complete. □

It is also true for the LP (5) that the null solution occurs if and only if condition (9) holds [4]. However, condition (9) was written in a slightly more complex form in [4, (10)]. For real-world classification problems, all k classes rarely have the same mean. Thus the null solution does not pose a computational difficulty from a practical standpoint.

3. PQM and partial gradient distribution. In this section we examine serial and parallel methods for solving the PQM (6). This problem may be solved serially by any unconstrained first-order optimization method. Our computational results, presented in §4, indicate that a quasi-Newton method [20, p. 2] was considerably faster than solving the corresponding linear program (5). Parallel approaches are attractive for machine learning problems because the problem size may be quite large and the problem may need to be solved many times in the course of a decision tree construction. We took advantage of the structure of the problem and applied the parallel gradient distribution (MCD-PGD) method [16] that is described below. We refer the reader to [16] for more details of MCD-PGD.

The PGD algorithm theorem is based on forcing function arguments. The definition of a forcing function is provided below. Some typical forcing functions are $\alpha\zeta$, $\alpha\zeta^2$, $\max\{\sigma_1(\zeta), \sigma_2(\zeta)\}$, $\min\{\sigma_1(\zeta), \sigma_2(\zeta)\}$, where $\sigma_1(\zeta)$ and $\sigma_2(\zeta)$ are forcing functions.

DEFINITION 3.1. Forcing function. *A continuous function σ from the nonnegative real line R_+ into itself such that $\sigma(0) = 0$, $\sigma(\zeta) > 0$ for $\zeta > 0$ and such that for the*

sequence of nonnegative real numbers $\{\zeta_i\}$:

$$\{\sigma(\zeta_i)\} \to 0 \text{ implies } \{\zeta_i\} \to 0$$

is said to be a forcing function on the sequence $\{\zeta_i\}$.

The following theorem describes the PGD used for this work. See [16] for the convergence proof of this theorem and other related algorithms.

THEOREM 3.1. PGD algorithm Theorem 1 [16, Corollary 3.2] *Let* $f \in C^1(R^h)$. *Start with any* $x_0 \in R^h$. *Having* x_i *stop if* $\nabla f(x_i) = 0$, *else compute* x_{i+1} *from directions* $d_i^\ell \in R^{h^\ell}$, $\ell = 1, \ldots, k$, $\sum_{\ell=1}^k h^\ell = h$, *as follows*:

Direction d_i^ℓ:

$$-\nabla_\ell f(x_i) d_i^\ell \geq \tau_\ell(\|\nabla_\ell f(x_i)\|), \qquad \ell = 1, \ldots, k, \tag{14}$$

where τ_ℓ *is a forcing function on* $\{\|\nabla_\ell f(x_i)\|\}$, $\ell = 1, \ldots, k$.

Asynchronous stepsize: *Choose* y_i^ℓ, $\ell = 1, \ldots, k$, *such that for* $\bar{\ell}$, *the complement of* ℓ *in* $\{1, \ldots, k, \}$:

$$f(x_i) - f(y_i^\ell, x_i^{\bar{\ell}}) \geq \mu_\ell(-\nabla_\ell f(x_i) d_i^\ell) \geq 0, \qquad \ell = 1, \ldots, k, \tag{15}$$

where μ_ℓ *is a forcing function on the sequence of nonnegative real numbers* $\{-\nabla_\ell f(x_i) d_i^\ell\}$ *for bounded* $\{d_i^\ell\}$, $\ell = 1, \ldots, k$.

Synchronization: *Find* x_{i+1} *such that*

$$f(x_{i+1}) \leq \min_{1 \leq \ell \leq k} f(y_i^\ell, x_i^{\bar{\ell}}). \tag{16}$$

Then, either $\{x_i\}$ *terminates at a stationary point* $x_{\bar{i}}$ *of* $\min_x f(x)$, *or for each accumulation point* $(\bar{x}, \bar{d})$ *of* $\{x_i, d_i\}$, $\bar{x}$ *is a stationary point of* $\min_x f(x)$.

In [16] a number of implementations of Theorem 3.1 were proposed including gradient descent and quasi-Newton directions and stepsizes such as the Armijo and minimization stepsizes. We shall use another implementation of Theorem 3.1 based on the following simple remarks. Instead of choosing y^ℓ, $\ell = 1, \ldots, k$, so as to satisfy the realizable inequalities (14) and (15), we take the best possible y_i^ℓ, that is:

$$f(y_i^\ell, x_i^{\bar{\ell}}) = \min_{x^\ell} f(x^\ell, x_i^{\bar{\ell}}), \qquad \ell = 1, \ldots, k. \tag{17}$$

Hence conditions (14) and (15) are satisfied for some τ_ℓ and μ_ℓ, $\ell = 1, \ldots, k$. Similarly for the synchronization step (16), take the best possible x_{i+1} over the affine hull of x_i and $(y_i^\ell, x^{\bar{\ell}})$, $\ell = 1, \ldots, k$. Hence (16) is satisfied. Consequently, we have the following parallel algorithm that we propose for our multicategory discrimination (MCD) problem. This algorithm can also be considered as a parallel variable distribution algorithm [10].

THEOREM 3.2. PGD algorithm Theorem 2. *Let* $f \in C^1(R^h)$. *Start with any* $x_0 \in R^h$. *Having* x_i *stop if* $\nabla f(x_i) = 0$, *else compute* x_{i+1} *as follows.*

Parallelization: *Find* $y_i^\ell \in R^{h^\ell}$, $\ell = 1, \ldots, k$ *such that*

$$(y_i^\ell, x^{\bar{\ell}}) \in \arg\min_{x^\ell} f(x^\ell, x_i^{\bar{\ell}}). \tag{18}$$

Synchronization: *Find* $x_{i+1} \in R^h$ *such that*

(19)

$$\begin{aligned} x_{i+1} &= \lambda_i^0 x_i + \lambda_i^1(y_i^1, x_i^{\bar{1}}) + \cdots + \lambda_i^k(y_i^k, x_i^{\bar{k}}) \\ &\in \arg \min_{\lambda^0, \lambda^1, \ldots, \lambda^k} f(\lambda^0 x_i + \lambda^1(y_i^1, x_i^{\bar{1}}) + \cdots + \lambda^k(y_i^k, x_i^{\bar{k}})). \end{aligned}$$

Then, either the algorithm terminates at a stationary solution $x_{\bar{i}}$ *of* $\min_x f(x)$ *or for each accumulation point* $\bar{x}$ *of* $\{x_i\}$, $\bar{x}$ *is a stationary point of* $\min_x f(x)$.

A natural way to apply Theorem 3.2 to PQM (6) is to let $x_i^\ell = (w_i^\ell, \gamma_i^\ell)$. This results in the following algorithm.

ALGORITHM 3.1. **MCD-PGD(MCD via PGD)** *Start with* $x_0 = (w_0, \gamma_0) \in R^{kn+k}$ *and define* $f(x) = f(w, \gamma)$ *as in* (6).

• *Stop if* $\nabla f(x_i) = 0$. • Parallelization: *For each class* $\ell = 1, \ldots, k$ *find* $y_i^\ell \in R^{n+1}$:

$$(y_i^\ell, x_i^{\bar{\ell}}) \in \arg\min_{x^\ell} f(x^\ell, x_i^{\bar{\ell}}). \tag{20}$$

Stop if $\nabla f(y_i^\ell, x_i^{\bar{\ell}}) = 0$.

• Synchronization:

$$\begin{aligned} x_{i+1} &= \lambda_i^0 x_i + \lambda_i^1 (y_i^1, x_i^{\bar{1}}) + \cdots + \lambda_i^k (y_i^k, x_i^{\bar{k}}) \\ &\in \arg \min_{\lambda^0, \lambda^1, \ldots, \lambda^k} f(\lambda^0 x_i + \lambda^1 (y_i^1, x_i^{\bar{1}}) + \cdots + \lambda^k (y_i^k, x_i^{\bar{k}})) \end{aligned}$$

• *Repeat.*

We used the following simple heuristic for choosing a starting point $x_0 = (w_0, \gamma_0)$, which consists of taking w_0^ℓ as the difference between the mean of class ℓ and the mean of all the points:

$$w_0^\ell = \frac{e^\ell A^\ell}{m^\ell} - \frac{\sum_{j=1}^k e^j A^j}{\sum_{j=1}^k m^j}, \quad \gamma_0^l = 0, \quad l = 1, \ldots, k. \tag{21}$$

The unconstrained convex minimization subproblems (20) and (21) were solved by using the quasi-Newton algorithm in the MINOS [20] optimization package.

Many variations of the direction and synchronization steps of Algorithm 3.1 are possible under Theorem 3.1. The algorithm presented was the best we found computationally. The algorithm is easily parallelized by distributing each of the subproblems (20) among k processors. The processors then synchronize once to share the results of the ℓ subproblems and the result of the synchronization step. We limited the number of iterations within the subproblems to the number of variables in the problem. This prevented one processor from spending too much time on one subproblem thus causing the other processors to be idle. We also relaxed the termination criteria slightly. The algorithm was halted if the gradient was sufficiently small (10^{-3}) or if the change in the objective function between major iterations was too small. Computational results in §4.2 show that the relaxation of the optimality condition did not adversely affect the quality of the solution found in terms of the number of points misclassified in the training and test sets.

We experimented with variations of the direction and synchronization steps. For example, the synchronization step (21) was replaced with a strong convex combination of the k points found in step (20) such as the average of the k points. This synchronization step was too conservative. The time per iteration was reduced but the number of iterations greatly increased. The final approach described above was adopted after a number of trials.

TABLE 1
Description of datasets used in experiments.

Dataset	Dimension	Classes	Total points
Wine	13	3	178
Iris	4	3	150
Glass	9	7	214
Image-s	19	7	210
Image-n	19	7	2310

4. Computational results. We conducted a series of computational experiments to investigate three questions: How does the linear program formulation (5) compare with the PQM formulation (6)? How does the serial MCD-PGD algorithm compare with a purely serial quasi-Newton algorithm? And how well does the MCD-PGD algorithm perform on a parallel machine? The serial experiments were performed on a DECstation 5000/125. The parallel experiments were performed on a Thinking Machines CM-5 parallel processor. The LP and quadratic subproblems were solved using the MINOS [20] package. Actual discrimination problems were used to compare the algorithms. These datasets are available via anonymous ftp (file transfer protocol) from the University of California-Irvine Repository of Machine Learning Databases and Domain Theories [19]. The wine recognition data [1], referred to as wine, uses the chemical analysis of wine to determine the cultivar. This wine set is piecewise-linear separable. Fisher's classical Iris identification problem [11], referred to as Iris, used physical attributes of Iris blossoms to determine the type of Iris. The Iris data is almost piecewise-linear separable. In the forensic glass identification data [9], referred to as glass, the chemical analysis of forensic glass is used to determine the origin of the glass. The glass data is not piecewise-linear separable. In the image segmentation problem [6], low-level real-valued image features are used to determine the image segment: sky, cement, window, brick, grass, foliage, or path. This image data was generated by the Vision Group at the University of Massachusetts. The image data is divided into two parts: a training set consisting of 210 points and a testing set consisting of 2310 points. We refer to the set of 210 points as image-s since it is piecewise-linear separable, and the set of 2310 points as image-n since it is not piecewise-linear separable. Table 1 lists the number of points, attributes, and classes contained in each of the datasets. Note that all the computational results are for a single piecewise-linear separator, not a multivariate decision tree consisting of many piecewise-linear separators.

4.1. Comparison of serial implementation of LP and PQM. We compared the LP formulation (5) and the new piecewise-quadratic minimization formulation (6) on the machine learning problems wine, Iris, glass, and image-s described above. The linear program (5) was solved serially using MINOS [20] and PQM (6) was solved serially by a quasi-Newton method employed by MINOS. Note that the goal of these problems is to construct a function for classifying future unseen points. Thus we used three criteria to evaluate the algorithms: the time to construct the function (the training time), the percent correctness on the training set, and the percent correctness on unseen points. We used 10-fold cross validation [14] to estimate these criteria. In 10-fold cross validation, nine-tenths of the points were used for training and one-tenth of the points were held out and tested on the resulting function. This is repeated 10 times, once for each one-tenth used as the testing set. The results of the training time, testing set accuracy, and training set accuracy were averaged over

TABLE 2
Comparison of serial implementation of LP (5) *and* PQM (6).

Training set accuracy

Dataset	Average accuracy (%)		t-test
	LP	PQM	p
Wine	100.0 ± 0.0	100.0 ± 0.0	1.0
Iris	98.8 ± 0.6	98.8 ± 0.6	1.0
Glass	76.3 ± 1.2	74.1± 2.0	0.0005
Image-s	100.0 ± 0.0	99.9 ± 0.0	0.34

Testing set accuracy

Dataset	Average accuracy (%)		t-test
	LP	PQM	p
Wine	89.4 ± 7.6	93.9 ± 7.1	0.02
Iris	94.7 ± 6.8	97.3 ± 4.7	0.10
Glass	60.8 ± 11.4	61.3 ± 13.1	0.74
Image-s	79.1 ± 0.1	85.3 ± 7.4	0.08

Training time

Dataset	Training time (secs)		t-test
	LP	PQM	p
Wine	14.3 ± 1.2	5.2 ± 1.5	< 0.00001
Iris	5.8 ± 0.7	0.4 ± 0.1	< 0.00001
Glass	231.2 ± 25.2	12.4 ± 25.1	< 0.00001
Image-s	610.5 ± 107.0	96.9 ± 13.1	< 0.00001

the 10 trials. This was performed on each of the above datasets. Results are not reported for Image-n, because the LP for Image-n exceeded the space limitations of the MINOS version used for testing. The training set accuracies, testing set accuracies, and training times are given in Table 2 together with their standard deviations. The accuracies are given in terms of percent correctness. Times are seconds of CPU time on a DECstation 5000/125. The p-value gives the significance of a paired t-test between the results for the LP (5) and the results of PQM (6). Low p values indicate a significant difference between the means of the results.

The results indicate that the PQM formulation is considerably superior with respect to training time, sometimes by as much as an order of magnitude. The training set accuracies for the LP and PQM formulations were virtually the same. However, the testing set accuracy for PQM was better than the LP results. No conclusive explanations for this difference were found. Further investigation is needed to determine the best choice of error formulation for good generalization (testing set accuracy). The training time was clearly faster for PQM. Thus PQM achieved a significant improvement in run-time performance even before parallelization was introduced.

4.2. Comparison of serially implemented MCD-PGD and quasi-Newton. In the second set of experiments, we compared the computational results of solving PQM (6) with a quasi-Newton method versus solving PQM with the MCD-PGD Algorithm 3.1 implemented on the DECstation/125 serial machine. In addition to the wine, Iris, glass, and image-s problems, the large image-n problem was added. Table 3 gives the training set accuracies, the testing set accuracies, and the training times for both algorithms on the five datasets.

The average training set and testing set accuracies were not significantly different

TABLE 3
Comparison of serial implementation of MCD-PGD *and quasi-Newton algorithms on DECstation* 5000/125.

Training set accuracy

	Average accuracy (%)		t-test
Dataset	PGD	Quasi-Newton	p
Wine	100.0 ± 0.0	100.0 ± 0.0	1.00
Iris	98.7 ± 0.5	98.9 ± 0.6	0.34
Glass	73.5 ± 1.0	73.6 ± 1.5	0.60
Image-s	100.0 ± 0.0	100.0 ± 0.0	1.00
Image-n	96.0 ± 2.9	96.5 ± 1.3	0.003

Testing set accuracy

	Average accuracy (%)		t-test
Dataset	PGD	Quasi-Newton	p
Wine	91.6 ± 3.9	91.1 ± 8.3	0.84
Iris	96.0 ± 4.5	96.7 ± 4.7	0.34
Glass	63.1 ± 11.1	63.1 ± 10.5	1.00
Image-s	86.7 ± 6.5	86.7 ± 7.0	1.00
Image-n	95.5 ± 1.2	95.4 ± 1.3	0.68

Training time

	Training time (secs)		t-test
Dataset	PGD	Quasi-Newton	p
Wine	9.4 ± 2.0	4.5 ± 1.5	0.00025
Iris	1.6 ± 0.5	0.43 ± 0.1	0.00002
Glass	33.4 ± 2.4	32.4 ± 9.9	0.75
Image-s	111.5 ± 24.9	138.2 ± 58.1	0.09
Image-n	1294.0 ± 132.1	1748.2 ± 1130.6	0.20

on any dataset. The relaxation of the optimality criterion discussed in §3 does not adversely affect the testing set accuracy of the solution on these problems. In practice, stopping before the objective function is exactly optimal (i.e., $\nabla f(x_i) = 0$) may improve generalization as well as as training time. In machine learning applications, requiring exact optimality can cause overfitting. For the backpropagation algorithm [24], one successful stopping criterion is to reserve part of the training set as a tuning set, and to stop the algorithm when the accuracy on the tuning set decreases [15, pp. 41, 42]. In the future we plan to investigate the use of such tuning sets to halt the algorithm.

The training times for the MCD-PGD and quasi-Newton algorithms were competitive. For small problems the quasi-Newton algorithm is clearly a better choice. However, for larger problems such as glass, image-s, and image-n, MCD-PGD did as well as and even better than quasi-Newton. Ignoring communication costs and idle time, this indicates that, for large problems, 100% speedup efficiency *may* be achieved using parallel computation. The next section investigates the actual speedup efficiency achieved by MCD-PGD on the CM-5 parallel machine.

4.3. Comparison of parallel implementation of MCD-PGD and quasi-Newton. For the final set of comparisons, we implemented Algorithm 3.1 on the CM-5 parallel processor. For a k-class discrimination problem, we used a parallel version of MCD-PGD on k nodes. For comparison we ran the quasi-Newton method on one node. We limited the investigation to the three datasets (glass, image-s, and image-n) that exhibited promising theoretical speedup in the above serial experiment.

TABLE 4
Comparison of MCD-PGD *and quasi-Newton algorithms on the* CM-5.

Training Time

Dataset	Classes k	Quasi-Newton Training time (secs)	MCD-PGD Training time (secs)	t-test p	Time reduction	Efficiency (%)
Glass	7	63.3 ± 24.0	17.7 ± 3.0	< 0.00001	3.6	51
Image-s	7	203.6 ± 81.7	32.1 ± 9.1	< 0.00001	6.3	91
Image-n	7	2470.0 ± 1113.1	704.0 ± 137.0	< 0.00001	3.6	50

The average computation time over the 10 cross-validation runs is reported in Table 4. There was a significant decrease in computation time using MCD-PGD over quasi-Newton. The speedup efficiency, that is, the ratio of time on one node divided by k times the time on k nodes, was 50–91%. The lower efficiency is primarily caused by segments of the algorithm that create idle time. The subproblems (20) solved in the parallelization step may take different amounts of computational time. The other processors remain idle until the last processor finishes. We tried to minimize this effect by limiting the number of iterations in the subproblems. The synchronization step (21) in Algorithm 3.1 causes processors to remain idle thus decreasing the efficiency. Two possible approaches to improve efficiency are: (i) use of a cheaper synchronization step and (ii) allowing each processor to do its own synchronization as soon as it finishes. The latter approach is suitable for a shared memory machine and would result in an asynchronous algorithm. These are directions for future work.

5. Conclusion. We have proposed an easily parallelizable formulation for the MCD problem that consists of minimizing a piecewise-quadratic function. This formulation is comparable in accuracy to previous LP formulations, but is considerably faster when implemented serially. We developed a parallel gradient distribution algorithm to minimize a piecewise-quadratic error function on both serial and parallel machines. The serial implementation holds the promise of a fast parallel implementation once idle and communication costs are minimized. The parallel implementation efficiencies of 50–91% are good and can be further improved via an asynchronous algorithm. Actual computation time was reduced on average by a factor of 4.5.

REFERENCES

[1] S. AEBERHARD, D. COOMANS, AND O. DE VEL, *Comparison of classifiers in high dimensional settings*, Tech. Report 92-02, Departments of Computer Science and of Mathematics and Statistics, James Cook University of North Queensland, Australia, 1992.

[2] K. P. BENNETT AND O. L. MANGASARIAN, *Neural network training via linear programming*, in P. M. Pardalos, ed., Advances in Optimization and Parallel Computing, North Holland, Amsterdam, pp. 56–67, 1992.

[3] ———, *Robust linear programming discrimination of two linearly inseparable sets*, Optim. Methods Software, 1 (1992), pp. 23–34.

[4] ———, *Multicategory separation via linear programming*, Optim. Methods Software, 3 (1993), pp. 27–39.

[5] L. BREIMAN, J. FRIEDMAN, R. OLSHEN, AND C. STONE, *Classification and Regression Trees*, Wadsworth Advanced Books and Software, Belmont, CA, 1984.

[6] C. E. BRODLEY AND P. E. UTGOFF, *Multivariate decision trees*, COINS Tech. Report 92-83, University of Massachussets, Amherst, MA, 1992; Machine Learning, to appear.

[7] R. O. DUDA AND H. FOSSUM, *Pattern classification by iteratively determined linear and piecewise linear discriminant functions*, IEEE Trans. Electronic Computers, 15 (1966), pp. 220–232.

[8] R. O. DUDA AND P. E. HART, *Pattern Classification and Scene Analysis*, John Wiley & Sons, New York, 1973.

[9] I. W. EVETT AND E. J. SPIEHLER, *Rule induction in forensic science*, Tech. Report, Central Research Establishment, Home Office Forensic Science Service, Aldermaston, Reading, Berkshire RG7 4PN, 1987.

[10] M. C. FERRIS AND O. L. MANGASARIAN, *Parallel variable distribution*, SIAM J. Optim., 4 (1994), pp. 815–832.

[11] R. A. FISHER, *The use of multiple measurements in taxonomic problems*, Annual Eugenics, 7 (II) (1936), pp. 179–188.

[12] K. FUKUNAGA, *Statistical Pattern Recognition*, Academic Press, New York, 1990.

[13] S. GALLANT, *Optimal linear discriminants*, in Proc. Internat. Conf. Pattern Recognition, IEEE Computer Society Press, 1986, pp. 849–852.

[14] P. A. LACHENBRUCH AND R. M. MICKEY, *Estimation of error rates in discriminant analysis*, Technometrics, 10 (1968), pp. 1–11.

[15] K. LANG, A. WAIBEL, AND G. HINTON, *A time-delay neural network architecture for isolated word recognition*, Neural Networks, 3 (1990), pp. 23–43.

[16] O. L. MANGASARIAN, *Parallel gradient distribution in unconstrained optimization*, Computer Sciences Tech. Report 1145, University of Wisconsin, Madison, 1993; SIAM J. Control Optim., to appear.

[17] J. MINGERS, *An empirical comparison of selection measures for decision tree induction*, Machine Learning, 3 (1989), pp. 319–342.

[18] T. S. MOTZKIN AND I. J. SCHOENBERG, *The relaxation method for linear inequalities*, Canadian J. Math., 6 (1954), pp. 393–404.

[19] P. M. MURPHY AND D. W. AHA, *UCI repository of machine learning databases*, Tech. report, Department of Information and Computer Science, University of California, 1992.

[20] B. A. MURTAGH AND M. A. SAUNDERS, MINOS 5.1 *user's guide*, Tech. Report SOL 83.20R, Stanford University, Stanford, CA, January 1987.

[21] N. J. NILSSON, *Learning Machines*, MIT Press, Cambridge, MA, 1966.

[22] J. R. QUINLAN, *Decision trees as probabilistic classifiers*, in Proc. 4th Internat. Workshop on Machine Learning, M. Kaufmann, ed., Los Altos, CA, 1987.

[23] ———, *Simplifying decision trees*, Internat. J. Man-Machine Studies, 27(1), 1987.

[24] D. E. RUMELHART, G. E. HINTON, AND R. J. WILLIAMS, *Learning internal representations by error propagation*, in Parallel Distributed Processing, D. E. Rumelhart and J. L. McClelland, eds, MIT Press, Cambridge, MA, 1986, pp. 318–362.

[25] P. E. UTGOFF AND C. E. BRODLEY, *An incremental method for finding multivariate splits for decision trees*, in Proc. 7th Internat. Conf. Machine Learning, M. Kaufmann, ed., Los Altos, CA, 1990, pp. 58–65.

[26] P. E. UTGOFF AND C. E. BRODLEY, *Linear machine decision trees*, COINS Tech. Report 91-10, University of Massachusetts–Amherst, 1991.

SIAM J. OPTIMIZATION
Vol. 4, No. 4, pp. 735–753, November 1994
© 1994 Society for Industrial and Applied Mathematics
004

AN EXTENSION OF THE DQA ALGORITHM TO CONVEX STOCHASTIC PROGRAMS*

ADAM J. BERGER†, JOHN M. MULVEY†, AND ANDRZEJ RUSZCZYŃSKI‡

Abstract. The diagonal quadratic approximation (DQA) algorithm is extended for the case of risk-averse utility and other nonlinear functions associated with stochastic programs. The method breaks the stochastic program into a sequence of smaller quadratic programming subproblems that can be executed in parallel. Each subproblem is solved approximately by means of a convex version of a primal-dual interior-point code (LOQO). Convergence of the distributed DQA method is discussed.

All communication takes place among neighboring processors rather than via a master routine leading to an efficient *distributed* implementation. Results with a realworld airline planning model possessing a convex objective, 155,320 linear constraints and 303,600 variables, show the DQA algorithm's efficiency. The interior point direct solver (convex-LOQO) is shown to solve moderate-size stochastic programs in a small number of iterations (under 50).

Key words. stochastic programming, decomposition, parallel computation, interior point methods.

AMS subject classifications. 90C15, 90C30, 90C25, 68Q22

1. Introduction.

Stochastic programming (SP) has become an important research activity with significant applications in engineering and management. Some prominent examples include the following: telecommunications [30], production and manufacturing [8], transportation [10], [25], finance [14], [22], and airline planning [2]. In these problems, the stochasticities and the decision environment are integrated within a single multistage modeling framework. Due to computational and other limitations, real-world SP applications have been limited to optimizing a linear objective function corresponding to expected value, i.e., stochastic linear programs.

As any decision analyst knows, this objective is appropriate for a risk-neutral decisionmaker who is making an operational decision [13]. Unfortunately, many applications involve significant consequences—monetary and otherwise—that require the inclusion of risk-averse objectives. The resulting stochastic optimization model possesses a convex objective function. The actual functional form depends upon the method selected for handling risk. Two popular approaches are the von Neumann–Morgenstern (VM) expected utility model and the mean-variance quadratic program, also known as Markowitz's portfolio model [17].

This paper describes two practical computational approaches for handling convex objectives within an SP model. The first approach, for small to moderate size examples, is an extension of the primal-dual interior-point algorithm [33]. The second method, for larger examples, is called convex-DQA, where DQA stands for diagonal quadratic approximation. Both approaches are natural extensions of previous work with stochastic linear programs. The research contributions occur at two levels. First, we have generalized the primal-dual interior-point algorithm, as implemented in Vanderbei's LOQO code, to handle convex and *separable* objectives. To accommodate the separable requirement, we factorize the VM functions in a way that is similar to

* Received by the editors September 7, 1993; accepted for publication (in revised form) April 22, 1994. This research was supported in part by National Science Foundation grant CCR-9102660 and U.S. Air Force Office of Scientific Research grant 91-0359.

† Department of Civil Engineering and Operations Research, Princeton University, Princeton, New Jersey 08544 (ajberger@emerald.princeton.edu, mulvey@macbeth.princeton.edu).

‡ International Institute for Applied Systems Analysis, A-2361 Laxenburg, Austria.

the factorization of the quadratic functions in previous research [34]. Next, we use convex LOQO as a subproblem solver in the DQA algorithm.

The DQA method is designed for SP problems that are too large for direct solvers such as LOQO. A primary advantage of the DQA method—solution via a totally distributed environment—is preserved in the convex-DQA algorithm.

The advantages of a distributed algorithm are several. First, the lack of a master routine means that the usual communication bottleneck is avoided since subproblems talk only with neighboring subproblems. Second, computational costs are minimized whenever there is a group of workstations (or even powerful PCs) that are connected over a broadband network such as an Ethernet. In most cases, the workstations are greatly underutilized while users read email, edit files, or write research papers. The distributed DQA algorithm has the potential to exploit this underutilized resource. Third, there may be computers on the network with unequal capabilities, i.e., some powerful processors and some weaker processors. The distributed algorithm should be able to allocate its work in a fashion so that all processors are equally busy and elapsed time is minimized.

The rest of the paper is organized as follows. Section 2 describes the SP problem and presents an overview of the convex-DQA algorithm. Convergence proofs are briefly discussed. Computational results are presented in §3. These preliminary results are quite promising, showing that the convex-LOQO and convex-DQA methods can solve SP problems with risk averse objectives. Last, §4 discusses some avenues for future research.

2. Convex-DQA algorithm overview.

2.1. Multistage stochastic programming models. In a multistage stochastic programming problem with time stages $t = 1, \ldots, T$ and scenarios $i = 1, \ldots, S$, the sequences of decisions $x_i(t)$, $t = 1, \ldots, T$, must satisfy two groups of relations. The first group describes the set of feasible actions for each t:

$$x_i(t) \in X_i(t), \qquad t = 1, 2, \ldots, T, \tag{1}$$

where $X_i(t) \subseteq R^{m_x}$ is a closed convex polyhedron. The second group describes the dynamics of the system and relates decisions from different time stages. In the simplest linear model they may read

$$D_i(t)x_i(t-1) + H_i(t)x_i(t) = b_i(t), \qquad t = 1, \ldots, T. \tag{2}$$

Here $D_i(t)$ and $H_i(t)$, $t = 1, \ldots, T$ are sequences of $m_b \times m_x$ matrices and $b_i(t)$, $t = 1, \ldots, T$, is a sequence of vectors in R^{m_b}. Obviously, the dimensions need not be the same for different t; we just use one m_x and one m_b for simplicity.

Finally, there is a convex cost function $f : R^{m_x T} \to R$

$$f(x_i) = f(x_i(1), x_i(2), \ldots, x_i(T)).$$

The objective is to minimize its expectation

$$F(x) = \sum_{i=1}^{S} p_i f(x_i), \tag{3}$$

where p_i, $i = 1, \ldots, S$ denote probabilities of scenarios ($p_i \geq 0$, $\sum_{i=1}^{S} p_i = 1$).

Scenario data

$$s_i(t) = (X_i(t), D_i(t), H_i(t), b_i(t))$$

associated with time stage t become known at t, so it is reasonable to make the decision $x_i(t)$ dependent on the information available up to time t, but not on information revealed after t. This is called *nonanticipativity*: for each t decisions $x_i(t)$ must be equal for all scenarios i that have common past and present.

The problem can be now stated as the problem of minimizing (3) subject to the constraints (1) and (2) and the nonanticipativity constraint: for all $i = 1, \ldots, S$ and any $t \in \{1, \ldots, T\}$

$$x_i(t) = x_j(t) \text{ if } s_i(\tau) = s_j(\tau) \text{ for } \tau = 1, \ldots, t. \tag{4}$$

In other words, decisions corresponding to scenarios that are indistinguishable up to time t should be equal.

Of course, we could have defined the problem in a more general way: infinitely many scenarios defined on some appropriate probability space. But even in our simple linear formulation, multistage stochastic programming problems are too difficult to be successfully solved for general distributions of the random data. Problems with finitely many scenarios are more amenable for computer solutions but many difficulties still remain.

First of all, one must note the remarkable size of the problem. If the scenarios introduced to the model are to reflect uncertainties that occur at successive time stages, then the number S of scenarios grows exponentially with the increase of the time horizon T. Even for relatively small, T the dimension of (1)–(4) may be so large that the whole problem will become intractable by direct solvers.

However, (1)–(4) have a very special structure. All scenarios can be arranged into a tree: common data for $t = 1$, some subgroups for $t = 2$, sub-subgroups for $t = 3$, and so on. The nonanticipativity constraints (4) require that our decisions have a similar tree-like structure: the first-stage decisions must be the same for all scenarios, because the future is not yet known, second-stage decisions must be equal within some subgroups of scenarios that are indistinguishable on the basis of information available so far, and so on. An example of such a tree for an 8-scenario problem is shown in Fig. 1.

This creates a number of possibilities for exploiting the structure and developing special methods for solving (1)-(4).

Existing computational methods for multistage stochastic programming problems can be divided into two main groups. First, there are versions of general purpose algorithms in which special features of stochastic problems are used to improve data structures and solution strategies [5], [11], [12], [16]. Second, we have a number of special *decomposition methods* that exploit the structure of the problem to split it into manageable pieces and coordinate their solution [35]. One can distinguish two classes: *primal* decomposition methods that work with subproblems assigned to time stages [4], [28] or even smaller subgroups of variables [24] and *dual* methods, in which subproblems correspond to scenarios [19], [22], [26].

We shall develop here the dual decomposition method proposed for linear multistage stochastic programs in [19] and further developed in [20]. Its main idea can be summarized as follows:

(i) formulate nonanticipativity as a sparse system of equations;
(ii) remove nonanticipativity by an augmented Lagrangian;

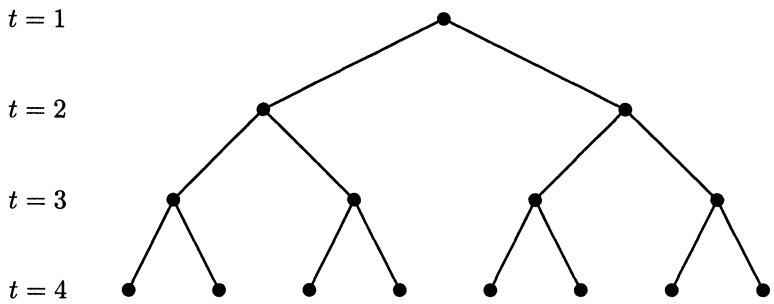


FIG. 1. *Scenario tree. Nodes at level t represent possible realizations of $s(t)$ and the paths from the root to the leaves represent possible sequences of data $s(1), s(2), \ldots, s(T)$.*

(iii) decouple the augmented Lagrangian using nonlinear Jacobi iterations;

(iv) solve Jacobi subproblems using quadratic approximations and an interior point method.

We also describe a totally distributed version of the method in which there is no coordinating unit and communication between subproblems is restricted to nearest neighbors.

2.2. Nonanticipativity. Clearly, the constraints in (4) describe a linear subspace of nonanticipative decisions in a massively redundant way. To develop an efficient numerical algorithm for solving multistage stochastic programming problems we need a description convenient for computation.

Let us define the *last common stage* of scenarios i and j by

$$t^{\max}(i,j) = \max\{t : \ s_i(\tau) = s_j(\tau), \ \tau = 1, \ldots, t\}. \tag{5}$$

We assume that scenarios are ordered in such a way that for every i scenario $i+1$ has the largest last common stage with i among all scenarios $j > i$:

$$t^{\max}(i, i+1) = \max\{t^{\max}(i,j) : \ j > i\}.$$

Scenarios in Fig. 2 are ordered in this way. A node at row t and column i represents $x_i(t)$ and horizontal dotted lines join nodes that should have identical decisions. It is easy to observe that with such an ordering the bundles of scenarios indistinguishable up to some time t form connected subsets of $\{1, \ldots, S\}$.

Next, for every scenario i and every time period t we define the *sibling* of the pair (i,t) as

$$\nu(i,t) = \begin{cases} i+1 & \text{if } t^{\max}(i, i+1) \geq t, \\ \min\{j : \ t^{\max}(i,j) \geq t\} & \text{otherwise.} \end{cases} \tag{6}$$

For the examples of Figs. 1 and 2 siblings of scenarios are shown in Table 1.

For every t the mapping $\nu(i,t)$ defines a permutation of $\{1, \ldots, S\}$, which maps bundles of indistinguishable scenarios onto themselves. It is easy to observe that $\nu(i,t) \neq i$, if the bundle of scenario i at stage t contains more than one member, while $\nu(i,T) = i$. The inverse permutation will be denoted by $\nu^{-1}(i,t)$.

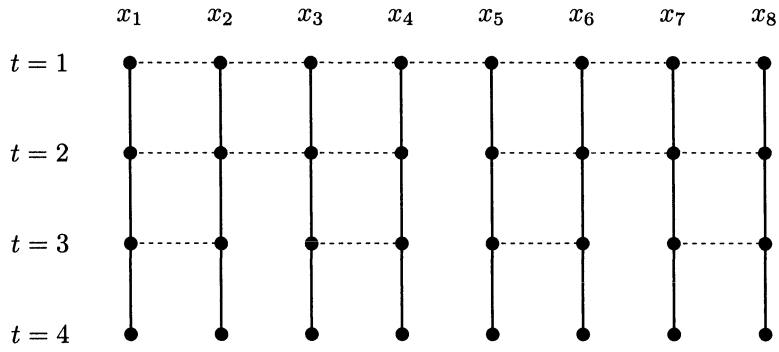


FIG. 2. *Sequences of decisions and nonanticipativity. Nodes connected by horizontal dotted lines must have identical decisions.*

TABLE 1
Siblings of scenario-time stage pairs.

Time stage	Scenario							
	1	2	3	4	5	6	7	8
1	2	3	4	5	6	7	8	1
2	2	3	4	1	6	7	8	5
3	2	1	4	3	6	5	8	7
4	1	2	3	4	5	6	7	8

Using the mapping $\nu(i,t)$ we can describe the nonanticipativity condition by the constraints

$$x_i(t) = x_{\nu(i,t)}(t), \quad i = 1,\dots,S, \quad t = 1,\dots,T-1. \tag{7}$$

There is still some redundancy in this set (we can remove one equation for each bundle), but we shall keep all equations (7) for convenience.

2.3. The augmented Lagrangian. For each scenario i the feasible set X_i defined by (1)–(2) is a closed convex polyhedron. The nonanticipativity constraints (4) can be put together into some joining constraints (7), which, although simple, are nonseparable and multidimensional. Thus, the whole problem has the following structure:

$$\min \sum_{i=1}^{S} p_i f(x_i) \tag{8}$$

subject to

$$x_i(t) = x_{\nu(i,t)}(t), \quad i = 1,\dots,S, \quad t = 1,\dots,T-1. \tag{9}$$

$$x_i \in X_i, \qquad i = 1,\dots,S. \tag{10}$$

The problem has a block diagonal structure with linking constraints, which suggests the application of the Dantzig–Wolfe decomposition method. Unfortunately, since the

nonanticipativity constraints (9) link almost all variables of the problem with some other variables from neighboring scenarios, the number of rows in the master problem would be extremely large. This makes the Dantzig–Wolfe method hard to implement for multistage stochastic programs.

An alternative dual approach to (8)–(10) is the use of the *augmented Lagrangian function*

$$\Lambda_r(x\ ,\pi) = \sum_{i=1}^{S} p_i f(x_i) + \sum_{i=1}^{S}\sum_{t=1}^{T-1}\langle \pi_i(t), x_i(t) - x_{\nu(i,t)}(t)\rangle + \frac{1}{2}r\sum_{i=1}^{S}\sum_{t=1}^{T-1} \| x_i(t) - x_{\nu(i,t)}(t) \|^2 . \tag{11}$$

Here $r > 0$ is a penalty parameter. The *multiplier method* (see, e.g., [3]) can be stated for (8)–(10) as follows.

ALGORITHM 1

1. For fixed multipliers π^k solve the problem

$$\min_{x\in X_1\times\ldots\times X_S} \Lambda_r(x, \pi^k). \tag{12}$$

Let $x^k = (x_1^k, x_2^k, \ldots, x_S^k)$ be the solution to (12).

2. If $x_i^k(t) = x_{\nu(i,t)}^k(t)$ for all i and t, then stop (optimal solution found); otherwise set for $i = 1, \ldots, S$ and $t = 1, \ldots, T-1$

$$\pi_i^{k+1}(t) = \pi_i^k(t) + r(x_i^k(t) - x_{\nu(i,t)}^k(t)), \tag{13}$$

increase k by 1 and go to step 1.

There are a number of general advantages of the augmented Lagrangian approach over usual dual methods: simplicity and stability of multiplier iterations and possibility of starting from arbitrary π^0 are among the most important ones.

A serious disadvantage of the augmented Lagrangian method is that (11) is not decomposable, so problem (12) cannot be split into S independent subproblems. In the next section we present a special approach in which we distribute step 1 of the multiplier method among the subproblems.

2.4. Diagonal quadratic approximation. Clearly, nonseparability of (11) is due to the quadratic terms

$$\| x_i(t) - x_{\nu(i,t)}(t) \|^2,$$

which contain cross-products $\langle x_i(t), x_{\nu(i,t)}(t)\rangle$.

Suppose that x belongs to a neighborhood of some reference point $\tilde{x}$. As noted in [19] and [20], we can then use the idea of [32] and approximate the products by separable expression

$$\langle x_i(t), x_{\nu(i,t)}(t)\rangle \approx \langle x_i(t), \tilde{x}_{\nu(i,t)}(t)\rangle + \langle \tilde{x}_i(t), x_{\nu(i,t)}(t)\rangle - \langle \tilde{x}_i(t), \tilde{x}_{\nu(i,t)}(t)\rangle.$$

Using this in (12) we can approximate step 1 of Algorithm 1 by S subproblems

$$\min_{x_i\in X_i}\left[\tilde{\Lambda}_r^i(x_i, \pi; \tilde{x}) = f(x_i) + \sum_{t=1}^{T-1}\langle \pi_i(t) - \pi_{\nu^{-1}(i,t)}, x_i(t)\rangle + \frac{1}{2}r\sum_{t=1}^{T-1}\{\| x_i(t) - \tilde{x}_{\nu(i,t)}(t) \|^2 + \| x_i(t) - \tilde{x}_{\nu^{-1}(i,t)}(t) \|^2\}\right]. \tag{14}$$

In other words, the augmented Lagrangian is minimized with respect to the variables associated with scenario i assuming that other variables are temporarily fixed at their values $\tilde{x}_j$ for all $j \neq i$, and this is done in parallel for each i.

The approximation point $\tilde{x}$ can be iteratively updated by the following Jacobi-type algorithm. Let us note that k is the iteration counter of the outer loop (Algorithm 1) and stays fixed during Algorithm 2.

ALGORITHM 2

1 Set $\pi = \pi^k$, $\tilde{x}^{k,m} = x^{k-1}$ and $m = 1$.

2 For $i = 1, \ldots, S$ solve (14) with $\tilde{x} = \tilde{x}^{k,m}$ obtaining new points $x_i^{k,m}$.

3 If $\| x_i(t) - \tilde{x}_i(t) \| \leq \epsilon$ for all i and $t < T$, where $\epsilon > 0$ is some prescribed accuracy, then stop; otherwise set for $i = 1, \ldots, S$, $t = 1, \ldots, T$

$$\tilde{x}_i^{k,m+1}(t) = \tilde{x}_i^{k,m}(t) + \alpha(x_i^{k,m}(t) - \tilde{x}_i^{k,m}(t)), \tag{15}$$

increase m by one and go to step 2.

By Theorem 1 of [29] convergence of the above algorithm is guaranteed for any value of the underrelaxation parameter in the interval

$$0 < \alpha < 1.$$

The general theory developed in [29] also allows us to estimate the speed of convergence of Algorithm 2. To do that, we need an additional assumption on the rate of growth of the augmented Lagrangian in the neighborhood of its minima. Let us define

$$\hat{\Lambda}_r(\pi) = \min_{x \in X} \Lambda_r(x, \pi)$$

and let $\hat{X}(\pi)$ be the set of minimizers of $\Lambda(x, \pi)$ in X. We assume that there is $\gamma > 0$ and $\delta > 0$ such that

$$\Lambda_r(x, \pi) - \hat{\Lambda}_r(\pi) \geq \gamma[\operatorname{dist}(x, \hat{X}(\pi))]^2$$

whenever $\operatorname{dist}(x, \hat{X}(\pi)) \leq \delta$. Then the method is convergent at a linear rate with the ratio

$$q = \frac{\alpha(1-\alpha)}{4r\gamma^{-1} + 1}. \tag{16}$$

The best estimate of the ratio in (16) can be obtained for $\alpha = \frac{1}{2}$.

The above results constitute a promising theoretical basis for developing an efficient practical method for general convex problems. An important question that needs to be answered is the way in which the subproblems (14) will be solved. We shall address it in §2.5.

2.5. Interior-point iterations. By DQA, the large problem (8)–(10) has been decomposed into a collection of nonlinear subproblems (14). To make the whole approach practical we need a fast and robust procedure for solving the nonlinear programming subproblems.

In [20] the original problem was linear resulting in quadratic programming subproblems with separable objectives. This suggested the application of interior point methods, which has proven to be a powerful tool for separable quadratic programming (see, e.g., [6], [7]). In our case we must deal with general convex objectives,

but it turns out that we can still apply the interior point methodology following the approach developed in [9] and [18].

We require an additional assumption about the separability of the function f:

$$f(x_i) = \sum_{j=1}^{J} f_j(x_{ij}) \qquad i = 1, \ldots, S \tag{17}$$

with all f_j's three times continuously differentiable.

Using a second-order Taylor's approximation of f near some x_0, we obtain a quadratic approximation of (14) in the form

$$\min {c_i}^T x_i + \frac{1}{2} x_i^T H_i x_i \tag{18}$$

subject to

$$A_i x_i = b_i, \tag{19}$$

$$x_i \geq 0. \tag{20}$$

Note that H_i is a diagonal matrix, and that the decision vector may be additionally expanded to convert linear and polyhedral constraints (1)–(2) into the standard form (19)–(20). We may add here that for many nonseparable problems a separable approximation can be developed using the transformation proposed in [34].

Since problem (18)–(20) is a separable quadratic program, identically as in [7], we can formulate its dual and use logarithmic barrier functions for nonnegativity constraints. Then the first-order optimality conditions for both primal and dual take on the form

$$A_i x_i = b_i, \tag{21}$$

$$A_i^T y_i + z_i - H_i x_i = c_i, \tag{22}$$

$$X_i Z_i e = \mu e, \tag{23}$$

where $\mu > 0$ is a barrier function coefficient, e is a vector of ones, y_i is the vector of multipliers associated with (19), z_i is the vector of nonnegative slack variables in the dual problem, and X_i and Z_i are diagonal matrices with diagonals x_i and z_i, respectively.

When the solution is strictly interior with respect to the nonnegativity constraints, Newton's method applied to (21)–(23) yields the following search directions:

$$\Delta z_i = -\rho(\mu) - X_i^{-1} Z_i \Delta x_i, \tag{24}$$

$$\Delta x_i = \Theta(A_i^T \Delta y_i - \rho(\mu)), \tag{25}$$

$$\Delta y_i = (A\Theta A^T)^{-1} A\Theta\rho(\mu), \tag{26}$$

where

$$\Theta = (H_i + X_i^{-1}Z_i)^{-1}$$

and

$$\rho(\mu) = (Z_i - \mu X_i^{-1})e - A_i^T y - z + H_i x_i + c_i.$$

The directions (24)–(26) can be used to update primal and dual variables; stepsize and barrier coefficients can be calculated by rules described in [7], [9], [15], [16]. After such a step, a new quadratic approximation (18)–(20) is calculated and the iteration continues. The convergence theory developed in [9] and [18] provides a firm ground for such a procedure. In this way we can proceed until the solution of (18)–(20) will be found. However, we may stop after a small number of iterations (or just one) and update the approximation point $\tilde{x}$ by (15). Also, the basic algorithm applies to the case in which there is only a single subproblem, whereby the method becomes a convex extension of the primal-dual interior point algorithm. The latter code we refer to as convex LOQO to distinguish it from the scenario decomposition approach.

2.6. A parallel distributed implementation. DQA allows great flexibility in the form of subproblem decomposition. A full decomposition would select every scenario as a separate unit, but it is often more desirable to group scenarios together into a subproblem, within which all nonanticipativity links are explicit in the constraint matrix. Each subproblem is associated with a separate process on a processor, but there may be one or many subproblems per processor. Communication and updating occurs between subproblems.

To clarify the DQA communication structure, we initially assume one scenario per subproblem (it may be helpful to think of each subproblem being solved on a different processor, thus one scenario per computer). Then it is clear that iterations of the interior point method can be carried out (for a fixed $\tilde{x}$) independently in each subproblem. It is also true for the algorithm for updating $\tilde{x}$ by (15).

Changes in $\tilde{x}$ cause changes in the objective function of other subproblems, as can be seen by (14), but interactions occur only between siblings at a given stage: $\tilde{x}_i(t)$ influences the objective function in scenarios $j = \nu(i,t)$ and $j = \nu^{-1}(i,t)$, and vice versa. Therefore, we derive a communication structure between subproblems in which messages need to be passed between nearest neighbors. For the example of Fig. 1 and Table 1 the communication structure is depicted in Fig. 3.

The multiplier iteration (13) can also be carried out in a distributed fashion. Indeed, when Algorithm 2 stops, we have $x_i(t) = \tilde{x}_i(t)$, so $x_i(t)$ is known to the sibling $\nu(i,t)$, while $x_{\nu(i,t)}$ is known to subproblem i. Thus, both i and $\nu(i,t)$ can calculate $\pi(i,t)$ by (13). Clearly, the same argument applies to the sibling $\nu^{-1}(i,t)$. Thus, all subproblems can update their objectives in a distributed fashion.

Table 2 shows an example of the more general case of multiple scenarios per subproblem. As mentioned above, only links between scenarios in different subproblems are used to form the augmented Lagrangian; links between scenarios in the same subproblem are treated explicitly in the constraint matrix. In this example, subproblem 1 contains scenarios 1–4, subproblem 2 contains scenarios 5 and 6, subproblem 3 contains scenario 7, and subproblem 4 contains scenario 8, and only the links shown in Table 2 need be coordinated. The resulting communication structure of subproblems is shown in Fig. 4. It could also be obtained by clustering subgroups of nodes of Fig. 3.

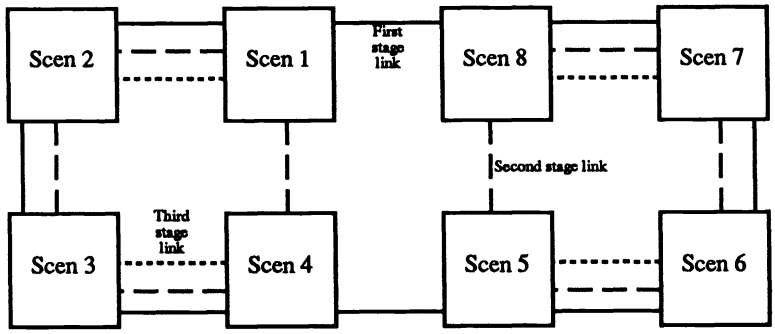


FIG. 3. *Communication structure; one scenario per subproblem.*

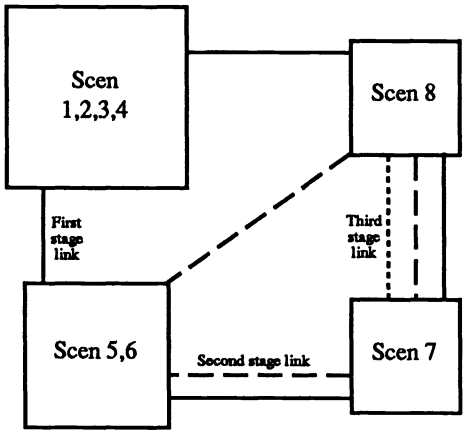


FIG. 4. *Communication structure; unequal subproblems.*

TABLE 2
Coordinated links by partial decomposition.

Time stage	I_1				I_2		I_3	I_4
	1	2	3	4	5	6	7	8
1	-	-	-	5	-	7	8	1
2	-	-	-	-	-	7	8	5
3	-	-	-	-	-	-	8	7
4	-	-	-	-	-	-	-	-

TABLE 3
Size of model constraint matrix as a function of number of scenarios.

Scenarios	Rows	Columns	Nonzeros
1	585	1,380	3,799
5	3,530	6,900	20,205
10	7,060	13,800	40,410
15	10,590	20,700	60,615
50	35,300	69,000	202,050
100	70,600	138,000	404,100
220	155,320	303,600	889,020

3. Numerical results. The DQA computational system described in [2] has been extended to address general convex objective functions based on the aforementioned concepts. As before, the code uses Bob Vanderbei's LOQO optimizer [33] for the interior point iterates. Communication among processors over the Ethernet is handled through PVM [1], [31], a general purpose interprocessor communication package. The code has been implemented on a variety of machine architectures, including Sun and NEXT workstations. The results detailed below employ the Silicon Graphics Indigo and Personal Iris machines.

Section 3 is divided into four parts. First we summarize the real-world test problems that form the basis of the experiments. Next, we present solution tests with the direct solver LOQO: solution time, computational limits, solution accuracy, and internal tactics. We evaluate the convex-DQA performance, both across a network of workstations (mainly testing parallelization capabilities) and on a single workstation, mainly testing parallelization capabilities. §3.4 discusses the tradeoff between accuracy and run time for convex-DQA.

3.1. Problem description. The experiments focus on a large real-world scheduling problem from the U.S Air Force. The model, called STORM, is used by the Air Mobility Command (AMC) to allocate military and civilian aircraft for peacetime cargo delivery on a monthly basis throughout the world. The model was originally developed for solution on the AT&T KORBX interior-point system and was converted to the GAMS modeling language.

AMC had formulated the model deterministically: aircraft routes and flying times, aircraft operation costs and specifications, cargo and frequency requirements, cargo handling costs, and Air Force base operating limits were all assumed known with certainty or based on past estimates. The DQA algorithm provides the capability to model stochasticities in any of these areas. Based on Air Force planners' recommendations, we concentrated on uncertainty in the demand for cargo to be shipped between destinations, which resulted in a two-stage stochastic linear program [20]. For the test cases that follow, each scenario of the STORM model requires 585 constraints, 1380 variables, and 3799 nonzero elements. The size of the resulting constraint matrices appears in Table 3 for different numbers of scenarios.

The first STORM objective is to minimize expected costs of the scheduling plan. This stochastic linear programming objective, however, puts no restriction on the variability of the plan across scenarios, or on the possibility of extremely poor results, as long as they are balanced out by very good results in other scenarios. To address this problem, we propose a utility function describing preferences across outcomes from all possible plans. A more robust objective would then be to maximize the

TABLE 4

Performance comparison for different objective functions in STORM model using convex LOQO. Convex objectives take about 3–4 times as long to solve, but only about 50% more iterations. Maximum problem size is about 28,000 variables.

		E[wealth] Linear		$E[U(w)]$ Exponential		$E[U(w)]$ Power	
Scenarios	Variables	Iter	CPU(s)	Iter	CPU(s)	Iter	CPU(s)
1	1,380	25	3.2	35	12	39	13
2	2,760	25	8.6	35	33	37	34
3	5,140	27	18	37	61	35	59
5	6,900	26	42	36	143	39	149
10	13,800	29	144	37	539	45	573
15	20,700	30	324	37	978	44	1,273
20	70,600	30	543	37	1,853	38	1,891

expected utility of final wealth, where wealth is defined as some constant (initial wealth) minus costs (see [21]). This extension gives rise to schedules that are less likely to change when implemented during the months as compared with the expected cost objective. Thus if costs in any scenario had been represented by $c_s^T x_s$ in the deterministic formulation, then we define wealth in scenario s as

$$w_s = w_0 - c_s^T x_s, \tag{27}$$

where w_0 is initial wealth, constant across scenarios. The robust objective becomes

$$\max E(U(w)) = \max \sum_{s=1,\ldots,S} p_s U(w_s), \tag{28}$$

where S is the total number of scenarios and p_s is the probability of scenarios s. Once constraint (27) is added to the subproblem the objective function in (28) takes on the form (17).

Three families of VM utility functions were tested:

(A) Risk Neutral $\quad U(w_s) = w_s,$

(B) Exponential Utility $\quad U(w_s) = -ae^{-bw_s},$

(C) Power Utility $\quad U(w_s) = aw_s^{\gamma},$

where w_0 is initial wealth, a,b, and γ are positive constants. Families (B) and (C) possess the desirable decreasing risk aversion property. Function (A) represents the expected value formulation of stochastic linear programming.

3.2. Convex LOQO. As a critical first step we modified the LOQO direct solver to handle convex objectives. Some typical results are illustrated in Table 4. For all three families of utility functions, LOQO handles problems of up to 20 scenarios on a single Silicon Graphics Indigo workstation, corresponding to an NLP problem with 14,120 rows and 27,600 columns. The convex objective problems took between 3–4 times as long to solve as the linear case, due primarily to increased matrix size and density. As typical with interior-point algorithms, the number of iterations remained relatively constant as problem size increased. Note convex problems require about 50% more iterations than the linear case.

A pertinent issue for the convex implementation is the number of interior point iterations that occur between quadratic approximation updates. In convex LOQO, we update the barrier coefficient (μ) and the quadratic approximation to the convex objective function (see (18)–(20)). For convex-DQA, we additionally update Lagrange multipliers for the dualized linking constraints (π) and the penalty parameter (r).

TABLE 5

Convex LOQO *solution time (in seconds) varying update frequency and utility functions of differing convexity. Updating the quadratic approximation every iteration yields good results. The highly nonlinear objective function takes slightly longer to solve.*

Iterations Between Updates	Convex Objectives: more ← **Curvature** → less				
1	155	141	142	141	137
2	148	143	142	141	140
3	197	136	143	140	143
4	157	143	139	138	145
5	200	182	139	139	144

For simplicity, it would be convenient to update the quadratic approximation every interior point iteration along with μ. We tested the viability of this strategy with the direct solver LOQO. In addition, we tested performance across utility functions of varying convexity of class (B).

The results in Table 5 yield two conclusions. First, updating the quadratic approximation every iteration came close to dominating the other updates tested and this became the strategy adopted for convex-DQA. Second, the LOQO direct solver handles all of the likely VM nonlinear objectives, though it was somewhat slower on the functions with greatest curvature (corresponding to highly risk averse utility).

Solution accuracy is based on the first-order optimality conditions (21)-(23). We maintain the same definitions previously employed in LOQO (see [34]).
Primal feasibility

$$PF = \frac{\| Ax - b \|_2}{\| b \|_2 + 1},$$

Dual feasibility

$$DF = \frac{\| A^T y + z - Hx - c \|_2}{\| c \|_2 + 1},$$

Duality Gap

$$DG = \frac{c^T x - b^T y + x^T H x}{| c^T x + x^T H x | + 1}.$$

Accuracy is defined as the maximum of {PF,DF,DG}. As evident in Table 6, there is a tradeoff between solution accuracy and run time. Improvement occurs rapidly until a duality gap of about 10^{-5}, after which many iterations are required for only a small improvement. For the results in this and the next section, the feasibility tolerances were set to 10^{-5} and the optimality tolerance (duality gap) to 10^{-4}.

3.3. Convex-DQA. Figure 5 summarizes the distributed DQA tests conducted across a network of four Silicon Graphics Indigo machines. For the runs up to and including 40 scenarios, the scenarios were divided into four equal subproblems with one subproblem assigned to each processor. Overhead (white portion of the bar) remains low until this point. Larger problems are solvable by assigning multiple subproblems per computer (Fig. 5), but overhead increases due to communication inefficiencies with the local Ethernet system at Princeton University.

There are also advantages to using the distributed decomposition algorithm DQA on a single machine. DQA solves problems of 35 scenarios on a single Indigo by

TABLE 6
Typical accuracy results for LOQO (12 *scenarios*, 15,000 *variables*). *Acceptable accuracy* (10^{-4}) *is attained quickly in* 38 *iterations.*

Iteration Number	Feasibility		Duality Gap	Solution Time	
	Primal	Dual		CPU	Elapsed
1	5×10^{2}	2×10^{2}	2×10^{1}	214	236
17	1×10^{-1}	9×10^{0}	1×10^{0}	442	467
22	1×10^{-2}	9×10^{-1}	2×10^{-1}	509	534
27	1×10^{-3}	8×10^{-2}	3×10^{-2}	576	602
32	1×10^{-4}	7×10^{-3}	2×10^{-3}	643	669
35	2×10^{-5}	9×10^{-4}	4×10^{-4}	683	709
38	2×10^{-6}	8×10^{-5}	3×10^{-5}	723	749
97	8×10^{-7}	5×10^{-5}	1×10^{-5}	1527	1562

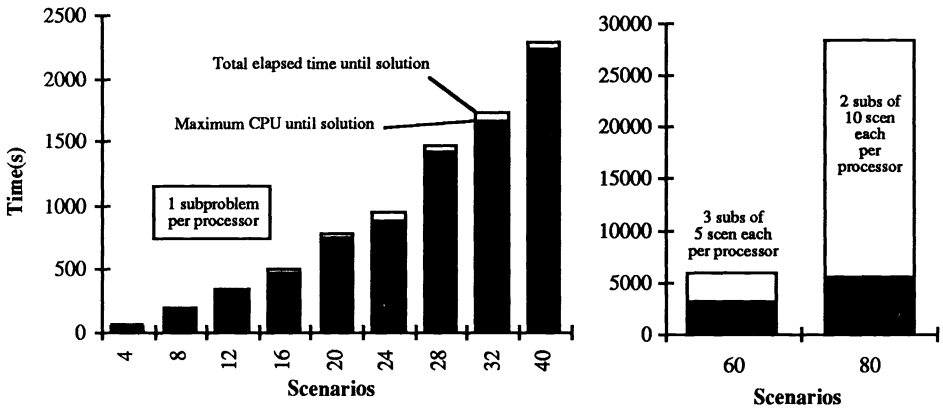


FIG. 5. DQA *on four processors*; STORM *model with exponential objective.*

breaking the problem down into seven subproblems of five scenarios each as compared with the 20 scenario maximum using the LOQO direct solver. Whereas factorizing a 35-scenario subproblem is too large for memory (32 megabytes), seven factorizations of five scenario subproblems is feasible. Figure 6 compares LOQO and DQA on a single machine, and DQA on four machines. The LOQO results refer to the exponential column of Table 4. Although DQA on a single machine does not perform as well as the direct solver, the machine's capability is expanded. We also see good speedup properties with four machine implementations. For example 20 scenarios requires 1,853 seconds on one machine but 733 seconds on four machines, which is about 2.5 times as fast, even though the single machine solution involves no decomposition.

Speedup properties were further tested via a network of Silicon Graphics Personal Iris workstations (PI). First we solved the 24 scenario problem with convex LOQO on the more powerful Silicon Graphics Iris 4D/480 computer at Princeton, which has processors of roughly the same speed as the PI machine but considerably more memory. Then, we tested the 24-scenario problem on a distributed network of 3, 4, 6, 8, and 12 PIs, with one subproblem on each processor (all subproblems of equal size). The results in Fig. 7 indicate good speedup properties; almost linear speedup for up to eight processors. Above this size Ethernet communication problems reduce viability of further decomposition.

Finally, the ability for DQA to handle larger configurations was tested on a network of 22 Silicon Graphics computers of differing capabilites. DQA solved a 220-

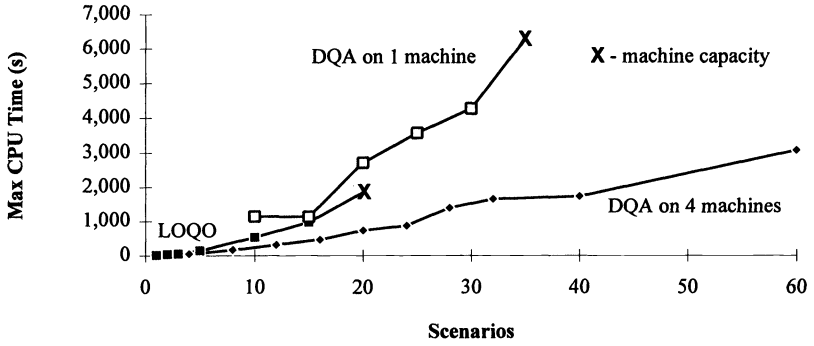


FIG. 6. *Comparison of different configurations. The decomposition method* DQA *allows a single machine to solve much larger problems than the direct solver* LOQO *can handle.* DQA *with four machines gives much faster solutions.*

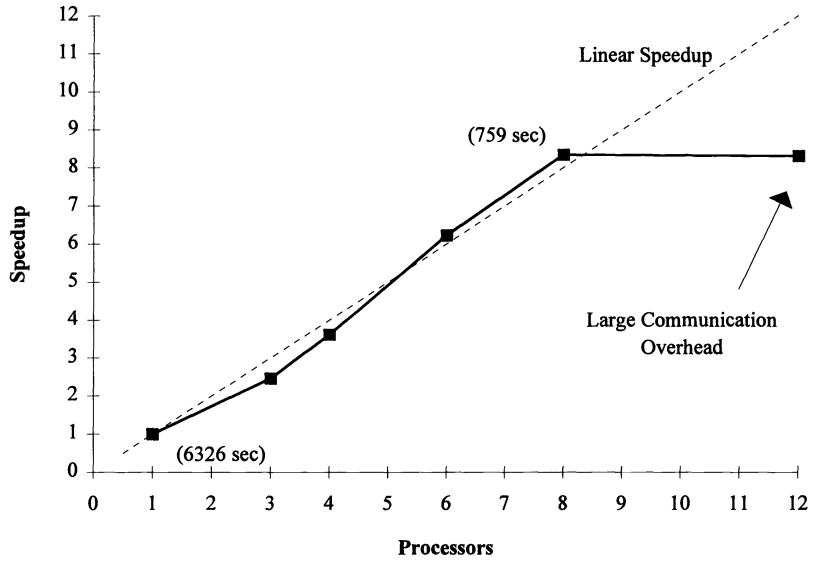


FIG. 7. *Parallel speedup; 24-scenario* STORM *model with exponential objective.*

scenario version of STORM (corresponding to an NLP with 155,000 variables and 303,000 constraints) by putting two subproblems of five scenarios each (a total of 44 subproblems) on every computer. CPU time is approximately 5300 seconds on the slower processors with an elapsed time of about 20,000 seconds. Almost three-quarters of the processing time involves communication overhead. With dedicated resources and a faster interprocessor communication mechanism, this overhead should diminish substantially. Regardless, DQA proved viable across a network of 22 processors, solving this large scale convex stochastic program to acceptable accuracy.

3.4. Accuracy. The major algorithmic difference in convex-DQA as compared with convex LOQO is due to the dualization of the nonanticipativity constraints. In convex-DQA, *nonanticipativity error* is defined as the 2-norm of the difference in the linked, nonanticipative variables across subproblems. For a direct comparison of accuracy between convex LOQO and convex-DQA, we will define primal feasibility in convex-DQA as the sum of the nonanticipativity error and the traditional error in

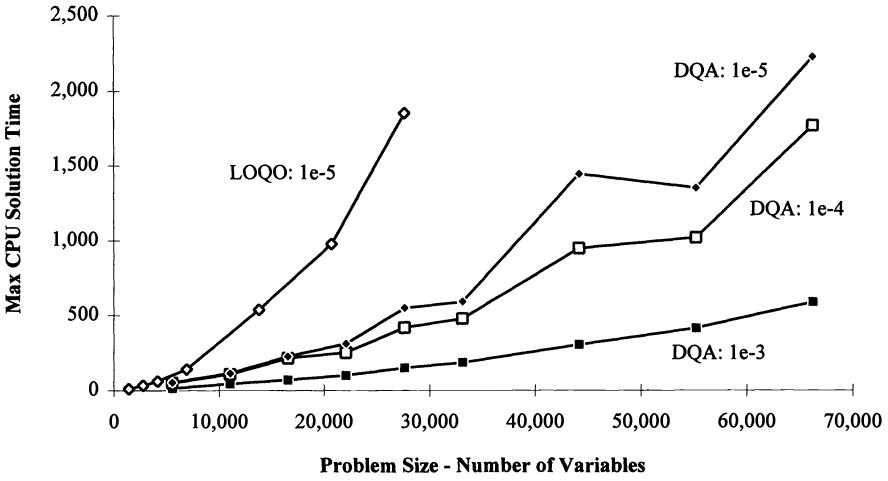


FIG. 8. *Convex-DQA accuracy test, four processors. Solution times can be very fast if very high accuracy is not needed.* LOQO *results provided for comparison.*

the primal feasibility constraints.

Performance and accuracy results are shown in Fig. 8 for three values of primal feasibility tolerances (10^{-3},10^{-4},10^{-5}) and for problems of varying size. These experiments took place on a network of four SGI Indigo machines. The results show that for larger problems, more time is required in convex-DQA for improvement than in LOQO, since an accuracy improvement requires another outer iteration, i.e., updating the Lagrange multipliers and penalty parameters and solving the resulting convex optimization problem. Comparable LOQO results are provided.

Figure 9 shows for a sample problem the extra time needed for DQA vs. LOQO for extra accuracy. Although DQA can not achieve accuracy as high as LOQO, it can attain very good accuracy ($< 10^{-5}$) in less time. Thus, the computational results confirm that *large* convex SP's can be solved with the convex-DQA algorithm. For smaller problems, the direct solver achieves high precision, but even for those problems, convex-DQA can be competitive providing that a high level of solution accuracy is not required. Likewise, convex LOQO should be a viable option for more powerful computers since the number of iterations is relatively small (less than 50).

4. Conclusions. This paper presents two approaches for solving stochastic programs possessing convex objectives. First, LOQO has been extended to handle separable, convex objectives. Second, for larger problems, the DQA algorithm has been modified in a similar way. Convex objectives are useful in several contexts: (1) to model risk averse decisionmaking, a critical feature when the outcomes of a modeling project are large vis-á-vis the decision maker's environment; (2) to deal with penalties for nonsatisfaction of second and subsequent stage constraints as defined in robust optimization [21]; and (3) to handle (dis)economies of scale. Since most decisionmakers display risk averse behavior, the convexity extension is critical to acceptance of the stochastic programs (models and algorithms) in practical applications.

What are some directions for future research? The first issue is to continue testing the convex LOQO and the convex-DQA methods to get a better idea of its pros/cons as compared with competing approaches for solving large *convex* stochastic programs. It is our experience that SP applications requires some preliminary analysis. For

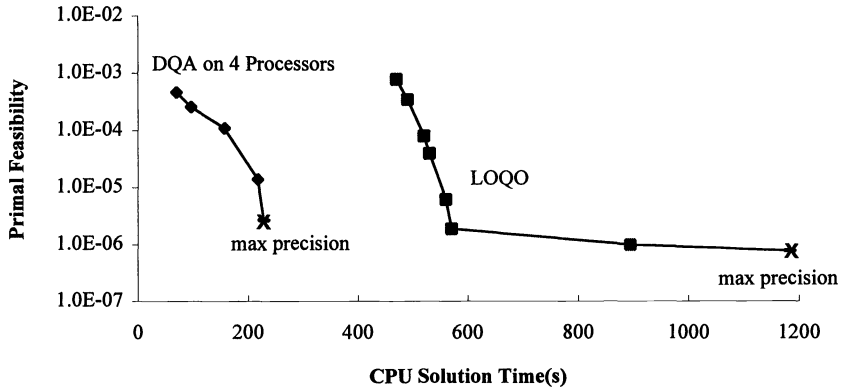


FIG. 9. *Accuracy vs. solution time comparison for* LOQO *and* DQA *(four machines), 15,000 variable problem.* DQA *is about three times as fast for a given primal feasibility level.* LOQO *is needed for primal feasibility* $< 10^{-6}$.

example, the generation of the scenarios is problem dependent and thereby difficult to generalize for a wide sprectum of problem domains. We plan to conduct computational tests in other application domains besides airline planning. Ideally, we would like to compare convex-DQA with methods such as the regularized decomposition algorithm and other likely competing methods [27].

Another research topic involves implementing the convex LOQO and convex-DQA algorithms with advanced hardware architectures. An obvious example is the tightly coupled (possibly massively parallel) CM-5 computer. We have begun prelimary work on this task and we plan to explore whether the CM-5 vector capabilities can be exploited in a cost-effective fashion. In previous work, we demonstrated that vector processors can be used to advantage with regard to speeding up the calculations when subproblems have an identical constraint structure, the A-matrix [2]. However, exploitation of the vector units requires further decomposition and more complicated software implementation.

We desire that stochastic programs become a practical tool for decisionmakers in engineering and management systems. To this end, we have successfully implemented stochastic programs in several planning areas. We plan to continue our implementation work using the convex LOQO and convex-DQA capabilities. These extensions allow risk averse behavior and related concerns to become an essential part of stochastic programming models.

Acknowledgment. The authors acknowledge Bob Vanderbei for his help with the LOQO computer code.

REFERENCES

[1] A. BEGUELIN, J. DONGARRA, A. GEIST, R. MANCHEK, AND V. SUNDERAM, *A Users' Guide to PVM Parallel Virtual Machine*, Tech. Report ORNL/TM-11826, Engineering Physics and Mathematics Div., Mathematical Sciences Section, Oak Ridge National Laboratory, Oak Ridge, TN, 1991.

[2] A. J. BERGER, J. M. MULVEY, AND A. RUSZCZYŃSKI, *Restarting strategies for the* DQA *Algorithm*, in Large Scale Optimization: State of the Art, W. W. Hager, D. W. Hearn, and P. M. Pardalos, eds., Kluwer, 1994, pp. 1–25.

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

[4] J. R. BIRGE, *Decomposition and partitioning methods for multistage stochastic linear programs*, Oper. Res., 33 (1985), pp. 989–1007.

[5] J. R. BIRGE AND D. HOLMES, *Efficient solution of two-stage stochastic linear programs using interior point methods*, Comput. Optim. Appl., 1 (1992), pp. 245–276.

[6] T. J. CARPENTER, *Practical Interior Point Methods for Quadratic Programming*, Ph.D. dissertation, Dept. of Civil Engineering and Operations Research, Princeton University, Princeton, NJ, 1992.

[7] T. J. CARPENTER, I. J. LUSTIG, J. M. MULVEY, AND D. F. SHANNO, *Separable quadratic programming via a primal-dual interior point method and its use in a sequential procedure*, ORSA J. Comput., 5 (1993), pp. 182–191.

[8] L. F. ESCUDERO AND P. V. KAMESAM, *On solving stochastic production planning problems via scenario modelling*, Tech. report, IBM T.J. Watson Research Center, Yorktown Heights, NY, 1993.

[9] A. S. EL-BAKRY, R. A. TAPIA, T. TSUCHIYA, AND Y. ZHANG, *On the formulation and theory of the primal-dual Newton interior-point method for nonlinear programming*, Tech. report TR92-40, Rice University, Houston, TX, 1993.

[10] L. F. FRANTZESKAKIS AND W. B. POWELL, *A successive linear approximation procedure for stochastic, dynamic vehicle allocation problems*, Transportation Sci., 24(1990), pp. 40–57.

[11] J. GONDZIO AND A. RUSZCZYŃSKI, *A sensitivity method for basis inverse representation in multistage stochastic linear programming problems*, J. Optim. Theory Appl., 74 (1992), pp. 221–242.

[12] E. R. JESSUP, D. YANG, AND S. A. ZENIOS, *Parallel factorization of structured matrices arising in stochastic programming*, SIAM J. Optim., 4 (1994), pp. 833–846.

[13] R. KEENEY AND H. RAIFFA, *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*, John Wiley, New York, 1976.

[14] M. KUSY AND W. ZIEMBA, *A bank asset liability model*, Oper. Res., 34 (1986), pp. 356–376.

[15] I. J. LUSTIG, R. E. MARSTEN, AND D. F. SHANNO, *On implementing Mehrotra's predictor-corrector interior point method for linear programming*, SIAM J. Optimization, 2 (1992), pp. 435–449.

[16] I. J. LUSTIG, J. M. MULVEY, AND T. J. CARPENTER, *Formulating stochastic programs for interior point methods*, Oper. Res., 39 (1991), pp. 757–770.

[17] H. M. MARKOWITZ, *Portfolio Selection: Efficient Diversification of Investments*, John Wiley, New York, 1959.

[18] R. D. C. MONTEIRO AND I. ADLER, *An extension of Karmarkar type algorithm to a class of convex separable programming problems with global linear rate of convergence*, Math. Oper. Res., 15 (1990), pp. 408–422.

[19] J. M. MULVEY AND A. RUSZCZYŃSKI, *A diagonal quadratic approximation method for large scale linear programs*, Oper. Res. Lett., 12 (1992), pp. 205–215.

[20] ———, *A new scenario decomposition methods or large-scale stochastic optimization*, Tech. report SOR 91-19, Dept. of Civil Engineering and Operations Research, Princeton University, Princeton, NJ, 1991; Oper. Res., to appear.

[21] J. M. MULVEY, R. J. VANDERBEI, AND S. A. ZENIOS, *Robust optimization of large scale systems*, Report 91–06–04, Decision Sciences Department, The Wharton School, University of Pennsylvania, Philadelphia, 1991; Oper. Res., to appear.

[22] J. M. MULVEY AND H. VLADIMIROU, *Stochastic network programming for financial planning problems*, Management Sci., 38 (1992), pp. 1642–1664.

[23] ———, *Solving multistage stochastic networks: an application of scenario aggregation*, Networks, 21 (1991), pp. 619–643.

[24] S. NIELSEN AND S. A. ZENIOS, *A massively parallel algorithm for nonlinear stochastic network problems*, Oper. Res., 41 (1993), pp. 319–337.

[25] W. B. POWELL, *A comparative review of alternative algorithms for the dynamic vehicle allocation problem*, in Vehicle Routing: Methods and Studies, B. L. Golden and A. A. Assad, eds., North-Holland, Amsterdam, 1988, pp. 249–291.

[26] R. T. ROCKAFELLAR AND R. J.-B. WETS, *Scenarios and policy aggregation in optimization under uncertainty*, Math. Oper. Res., 16 (1991), pp. 1–23.

[27] A. RUSZCZYŃSKI, *A regularized decomposition method for minimizing a sum of polyhedral functions*, Math. Programming, 35 (1986), pp. 309–333.

[28] ———, *Parallel decomposition of multistage stochastic programs*, Math. Programming, 58 (1993), pp. 201–228.

[29] ———, *Augmented Lagrangian decomposition for sparse convex optimization*, working paper WP-92-75, IIASA, Laxenburg, 1992; Mathematics of Oper. Res., to appear.

[30] S. SEN, R. DOVERSPIKE, AND S. COSARES, *Network planning with random demand*, University

of Arizona Report, Tucson, December, 1992.

[31] V. S. SUNDERAM, PVM: *A framework for parallel distributed computing*, Concurrency: Practice and Experience, 2 (1990), pp. 315–339.

[32] G. STEPHANOPOULOS AND W. WESTERBERG, *The use of Hestenes' method of multipliers to resolve dual gaps in engineering system optimization*, J. Optim. Theory Appl., 15 (1975), pp. 285–309.

[33] R. J. VANDERBEI, LOQO *User's Manual*, Tech. report SOR 92-5, Dept. of Civil Engineering and Operations Research, Princeton University, Princeton, NJ, 1992.

[34] R. J. VANDERBEI AND T. J. CARPENTER, *Symmetric indefinite systems for interior-point methods*, Math. Programming, 58 (1993), pp. 1–32.

[35] R. J.-B. WETS, *Large scale linear programming techniques*, in Numerical Methods in Stochastic Programming, Yu. Ermoliev and R.J.-B. Wets, eds., Springer-Verlag, Berlin, 1988, pp. 65–94.

SIAM J. OPTIMIZATION
Vol. 4, No. 4, pp. 754–776, November 1994
© 1994 Society for Industrial and Applied Mathematics
005

PROBLEM FORMULATION FOR MULTIDISCIPLINARY OPTIMIZATION*

EVIN J. CRAMER†, J. E. DENNIS, JR.‡, PAUL D. FRANK†,
ROBERT MICHAEL LEWIS‡, AND GREGORY R. SHUBIN†

Abstract. This paper is about multidisciplinary (design) optimization, or MDO, the coupling of two or more analysis disciplines with numerical optimization.

The paper has three goals. First, it is an expository introduction to MDO aimed at those who do research on optimization algorithms, since the optimization community has much to contribute to this important class of computational engineering problems. Second, this paper presents to the MDO research community a new abstraction for multidisciplinary analysis and design problems as well as new decomposition formulations for these problems. Third, the "individual discipline feasible" (IDF) approaches introduced here make use of existing specialized analysis codes, and they introduce significant opportunities for coarse-grained computational parallelism particularly well suited to heterogeneous computing environments.

The key distinguishing characteristic of the three fundamental approaches to MDO formulation discussed here is the kind of disciplinary feasibility that must be maintained at each optimization iteration. Other formulation issues, such as the sensitivities required, are also considered. This discussion highlights the trade-offs between reuse of existing software, computational requirements, and probability of success.

Key words. constrained optimization, multidisciplinary design optimization, optimal design, computational engineering

AMS subject classifications. 93, 65, 90

1. Introduction.

This paper is about multidisciplinary (design) optimization, or MDO. By MDO we mean the coupling of two or more analysis disciplines with numerical optimization. We expect MDO to have a tremendous economic impact in industry by shortening the design cycle and enabling the design of better systems and products at lower cost.

Mathematical programmers have much to contribute, especially through the advancement of modern algorithms for nonlinear and integer programming and their introduction into the MDO community. However, today's mathematical programming algorithms are not sufficient for many industrial MDO problems. To help the algorithm designer understand the kind of frontiers that must be pushed back, we carry throughout the paper the example of aeroelastic design. This enables us to give a context to research directions we hope will be followed. Another reason to include this rather lengthy example is to give a familiar, albeit simplified, context for the MDO community in which to understand and judge our abstraction and notation for the MDO problem.

Due to the extreme complexity of most MDO problems, we believe it is necessary to focus on problem formulation methods and their interdependence with nonlinear programming algorithms. In this paper we present a new abstraction of the MDO

* Received by the editors September 2, 1993; accepted for publication (in revised form) April 6, 1994

† The Boeing Company, P.O. Box 24346, Mail Stop 7L-21, Seattle, Washington, 98124-0346.

‡ Department of Computational and Applied Mathematics and Center for Research in Parallel Computation, Rice University, P.O. Box 1892, Houston, Texas, 77251-1892 (dennis@ariel.rice.edu). This work was supported by State of Texas contract 1059, Air Force Office of Scientific Research grants F49629-92-J-0203 and F49629-9310212, Department of Energy grant DE-FG005-86ER25017, and National Science Foundation cooperative agreement CCR-9120008.

problem, and we use it to examine alternative ways to formulate MDO problems. We do not concentrate here on the application of mathematical programming algorithms to MDO. The question of formulating MDO problems is a major topic in the engineering literature on MDO (e.g., [16]).

In part, the genesis of the ideas given here was to find ways to exploit parallel computation in nonlinear programming. We turned to reformulating the problems because, in our opinion, the design of parallel algorithms for general nonlinear programming has not been very successful. The new IDF formulations we suggest here have the advantage of a coarse-grained parallelism naturally suited to a heterogeneous computing environment. We hasten to point out that parallelism is not the only motivation for this formulation; indeed, even without parallelism, we expect IDF to be an attractive approach for solving MDO problems.

In §2, we present aeroelastic optimization as an example to provide intuition into the concepts presented in the rest of the paper. In §3, we present notation and definitions describing our abstract model of the multidisciplinary analysis and optimization problem. This model is used in §4 to discuss multidisciplinary analysis, the attainment of feasibility for MDO. Section 5 is a discussion of the three main formulations for MDO including one hinted at in other work, but stated explicitly here. Section 6 discusses the derivative requirements for MDO. Section 7 discusses some issues related to choosing a formulation, and states some concluding remarks. Section 8 presents a simple example of the different MDO formulations discussed in this paper.

The contents of this paper represent an abstraction and generalization of more specific material presented in [3].

2. Example. Aeroelastic optimization. A specific problem is very useful in thinking about MDO. For us the model problem is aeroelastic optimization. We use this example to define some terms, and throughout the text we refer to this example to illustrate the model and the various problem formulations. However, the model and formulations discussed in this paper are meant to apply to general MDO problems.

In static aeroelasticity we consider a flexible wing of an aircraft in steady flight. The air rushing over the wing causes pressures to be imposed on the wing, which causes the wing to deflect and change shape. This change in wing shape in turn causes the aerodynamic pressures to change. In static aeroelasticity, we assume that these physical processes reach an equilibrium.

The aeroelastic system in equilibrium is shown in Fig. 1. The two *analysis disciplines* involved are aerodynamics (D_1) and structures (D_2). The computational problems for these disciplines are generally solved by individual *analysis codes*, say, a finite difference computational fluid dynamics (CFD) code for aerodynamics (A_1) and a finite element code for structures (A_2).

It is important to note that Fig. 1 and others like it portray a purely static view of the relationships between the components and do not imply a sequence of calculations.

Suppose that the structures code has been given input parameters specifying the wing's structure, and that both the aerodynamics and structures code have been given input specifying the undeflected wing shape. The aerodynamics code takes as an additional input the wing deflections (M_{12}) and produces as output the pressures (and velocities, etc.) (U_1) on the wing surface. The structures code takes as an additional input the load on the wing (M_{21}) and produces as output the deflections (and stresses, etc.) (U_2) of the wing. We say that we have *single discipline feasibility* for aerodynamics when the CFD code (A_1) has been executed successfully and solved for

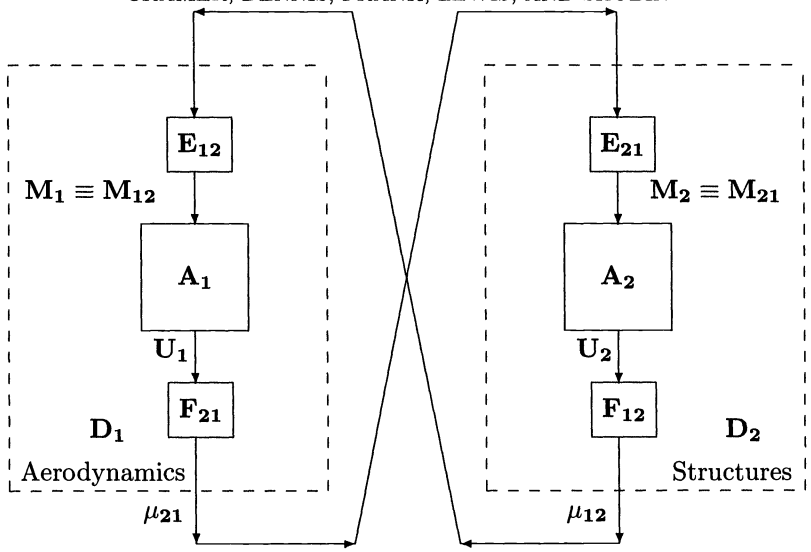


FIG. 1. *Aeroelastic system.*

the pressures, given an input shape. Similarly, we have single discipline feasibility for structures when the structures code (A_2) has successfully solved the structural analysis equations to produce deflections, given some input forces. Thus, "feasibility" for a single discipline means that the equations the discipline code is intended to solve are satisfied. Maintaining feasibility with respect to any user-specified design constraints is not an issue in this discussion, although it certainly would be an issue for the optimization algorithm that one might choose to apply to the nonlinear programming problem resulting from one of the formulations suggested here.

Continuing with the aeroelastic example, we note that the two analysis codes solve their problems on different computational grids (discretizations) and interact only at a specific interface. We accommodate this by following each analysis map, e.g., (A_1), by a map (F_{21}) that represents something like a spline fit to the grid values generated, and preceding the following analysis code by a map (E_{21}) that represents something like a spline evaluation to generate values at points needed by that analysis code, e.g., (A_2). Any additional computations required, such as converting pressures from aerodynamics into forces for structures, are assumed to be associated with either a fit or an evaluate routine, as appropriate. It will also be convenient to view a discipline as the analysis code together with all the E and F codes used to obtain its input and provide its output to other disciplines; this notion is depicted by the dashed boxes in Fig. 1.

We call all of the maps $E \circ F \triangleq G$ *interdisciplinary mappings.* They represent the coupling between disciplines and play a key role in MDO. We tacitly assume that each instance of an E or an F takes inputs from a single discipline and sends outputs to a single discipline. One way to handle instances of E or F that have more complex communication is to treat the subject mapping as a new "discipline." Note that the data passed between the disciplines in Fig. 1 may be considered "compressed" if μ_{21} and μ_{12} are much smaller vectors than U_1 and U_2, respectively. This would happen if, for example, the μ vectors represented coefficients of fitting functions with the U vectors as data. We call this "reducing the interdisciplinary bandwidth." Note that

an approximation would be made in such a fitting operation. As shown later, this data compression can be used to reduce the dimension of the optimization problem in certain formulations. These interdisciplinary mappings could also be implemented to provide a common interface between codes rather than the "all pairs" abstraction we are using here for simplicity.

A *multidisciplinary analysis* is achieved when:

1. we have single discipline feasibility in aerodynamics and in structures; and
2. the input to each corresponds to the output of the other via the interdisciplinary mappings.

We call this situation *multidisciplinary feasibility* and it corresponds to the simulations in Fig. 1 being in equilibrium, by which we mean that none of the values of the variables change upon execution of all mappings shown in Fig. 1 without regard to order. We discuss obtaining multidisciplinary feasibility in §4.

It is possible to have single discipline feasibility in both aerodynamics and structures (we call this *individual discipline feasibility*) and not have multidisciplinary feasibility. This occurs if the equations in each code are satisfied, but the input to one discipline does not correspond to the output of the other. This key observation plays an important role later when we present the individual discipline feasible (IDF) formulations for MDO.

We next add optimization to the aeroelastic example. Aerodynamic optimization combines the single analysis discipline aerodynamics with optimization. The *design variables* would typically be some parameters, say, spline coefficients, defining the wing's shape. The *objective function* might be drag, or some measure of closeness to some specified pressure distribution. There may be *design constraints* to prohibit undesirable wing shapes or bad aerodynamic flows. Similarly, structural optimization combines structures and optimization to minimize the structural weight by changing the size of structural components, subject to stress constraints. In aeroelastic optimization (the combination of aerodynamics, structures, and optimization) we generally have both *aerodynamic design variables* (shapes) and *structural design variables* (sizes and perhaps shapes). In our model, we lump all the design variables into a single vector variable X_D.

There is no consensus concerning the formulation of the aeroelastic optimization problem. Some logical choices are to minimize weight, subject to the constraint that drag be acceptably small, or to minimize drag, subject to weight being acceptably small. Alternatively, minimizing a combination of drag and weight might be appropriate. Ultimately, however, the aeroelastic behavior of the aircraft needs to be tied to some overall aircraft performance measure like direct operating cost. This situation, with a set of conflicting objectives, is to be expected in MDO because engineers in each discipline will probably have formulated their own objectives for the design. We will not consider the matter further in this paper, but the reader will note a goal programming approach to nonconvex multiobjective optimization in these comments (see [27]).

To appreciate the trade-offs between the various formulations of MDO problems presented later, it is necessary to know something about the size and difficulty of the underlying analysis disciplines. Obviously these factors depend on the problem, but we assume that the problem is sufficiently complex so that it cannot simply be overwhelmed with computing power. For example, a practical aeroelastic optimization for a three-dimensional configuration will involve a computational fluid dynamics code that takes hours of supercomputing time to execute a single analysis. The structural

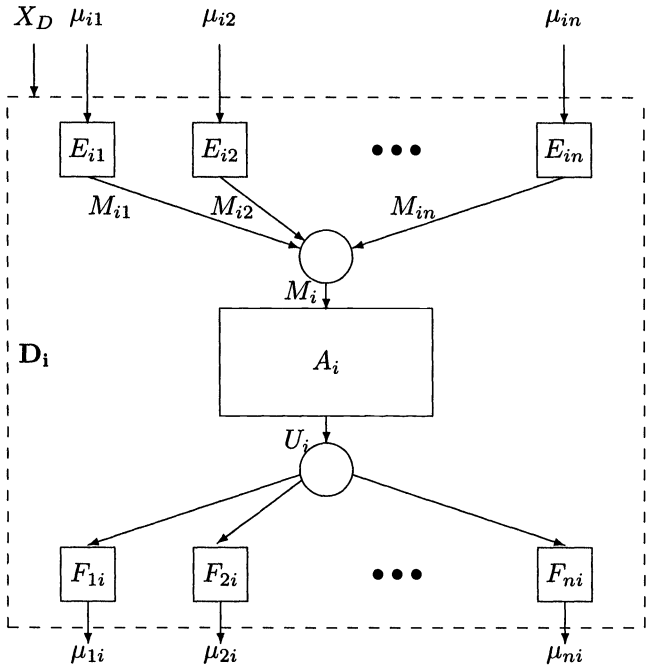


FIG. 2. *One of many disciplines.*

analysis code will typically be less costly, but may take a significant fraction of an hour. For either code the amount of computing time is acceptable for engineering analysis. However, many formulations of MDO require tens to hundreds of such executions; thus the impetus for MDO formulations requiring less computational work and the need to employ parallel computing even for the cheaper methods. A discussion of further considerations in choosing a formulation is postponed until §7.

3. A framework for describing MDO problems. In this section, we generalize the two discipline aeroelastic MDO examples to an abstraction for reasoning about general MDO problems. Figure 2 shows the data flow for a single discipline of a many-discipline version of Fig. 1.

In our notational convention, X denotes the vector of variables controlled by the optimizer. This is useful because it allows one to immediately identify what is, and what is not, an optimization variable. The original design variables X_D are always components of X, but in some formulations, X includes other variables as well. These other optimization variables may be thought of as surrogates for the quantities appearing as subscripts on X. For example, X_{U_i} is an optimizer-controlled surrogate for U_i. Constraints in the optimization problem are denoted by $C(X)$. The original design or system constraints $C_D(X)$ are always components of C, but C can include other constraints.

The notation $\partial C/\partial X$ represents the Jacobian matrix of C with respect to X. Thus, $[\partial C/\partial X]_{rs} = \partial C_r/\partial X_s$ and its rth row is the transpose of the gradient vector of constraint C_r.

An important convention is the way we use subscripts. When a quantity has double subscripts, the order indicates information flow as in "to–from." For example,

denote a generic ith single discipline by D_i. Then information meant to pass *to* D_i *from* D_j will be subscripted ij. It is useful to think of a discipline D_i as a grouping of communication and analysis codes. In terms of the example, the structural analysis code might have an "evaluator" code to provide loads where they are needed for the structural analysis. It might also have a "fitter" code to compress its output for communication to other disciplines. To avoid even more complexity, we allow these routines to pass some variables, like X_D, directly through; this has the effect of making X_D globally accessible while respecting the structure we wish to abstract. Constants needed for an analysis are assumed to reside where required.

We use the convention that arguments to the left of a semicolon are inputs to a function of a vector variable, and those to the right are the dependent variables to be determined by an equation involving the function.

3.1. Analysis inputs, equations, and outputs.

$M_i \triangleq$ Inputs to the analysis code A_i of discipline D_i. Components M_{ij} of M_i are inputs to analysis code A_i needed from discipline D_j. The long vector M comprises the subvectors M_i, for every i. (The M is mnemonic for "multidisciplinary data." A_i is to be executed with M_i and design parameters X_D as inputs.)

$n_{M_i} \triangleq$ The total number of interdisciplinary inputs to A_i, i.e., the length of the vector M_i.

$A_i \triangleq$ The analysis mapping of the form $U_i = A_i(X_D, M_i)$ from the inputs X_D, M_i of an analysis discipline to the outputs U_i. We often use the expressions "analysis solver" or "analysis code" to describe the computer program that implements the mapping. Much effort and talent have gone into developing these codes, so there are serious advantages to formulations that preserve their integrity.

$U_i \triangleq$ Quantities for which A_i solves internally when executed in D_i. These could include pressures, velocities, stresses, etc. As above, U denotes the vector of analysis discipline variables computed in a given formulation by solving the complete set of analysis discipline equations.

$n_{U_i} \triangleq$ Total number of analysis quantities, such as pressure, stresses, etc., associated with discipline i. For example, an analysis code A_i solves n_{U_i} equations for n_{U_i} analysis unknowns.

$W_i \triangleq$ Residual function of equations solved in D_i by A_i to compute the analysis variables U_i. These equations take the form $W_i(X_D, M_i; U_i) = 0$. We remind the reader that the variables to the left of the semicolon represent inputs to the system, while those to the right are the outputs (the variables for which A_i solves). There are n_{U_i} of these residuals.

3.2. Interdisciplinary mappings.

$G_{ij} \triangleq$ Mapping *to* the inputs required for analysis code A_i *from* the output of A_j. For example, G_{ij} could be the mapping of the pressures on the aerodynamic grid to the loads on the structures grid. The functional form of this mapping is $M_{ij} = G_{ij}(X_D, U_j)$, where the G_{ij} are the composition of two mappings $E_{ij} \circ F_{ij}$ given below. Sometimes there will be no input to A_i from A_j. Our convention for this is to set $G_{ij} = 0$.

$F_{ij} \triangleq$ Mnemonic for "fit." Mapping from the analysis variables from A_j to the outputs of D_j needed to produce input to D_i. For each i, j, F_{ij} has the role of transforming U_j for use in discipline i. This transformation will often involve a data

compression to reduce the communication bandwidth between disciplines. For example, $\mu_{ij} = F_{ij}(X_D, U_j)$ could map the pressures computed by aerodynamic analysis to the coefficients of a spline surface approximation to the pressures or to coefficients for a fit to the load induced on the wing by those pressures. There are n_{ij} such vector functions. Some may be identity mappings, and some may be zero mappings.

$\mu_i \triangleq$ Inputs to D_i from other disciplines. Components μ_{ij} of μ_i are sent by the fitter of D_j to the evaluator of discipline D_i. Our convention is that μ_{ij} may be just a compression of U_j by F_{ij} which will be transformed into M_{ij} by E_{ij}. Alternatively, μ_{ij} may be the product of a more complicated transmogrification. The symbol μ is mnemonic for "M or U." It is intended to reflect the nature of μ as a surrogate for M or U depending on the particular pair F_{ij}, E_{ij}. The vector μ is the block vector of all block vectors μ_i for every i.

$n_{\mu_i} \triangleq$ The total number of inputs μ_{ij} to D_i. That is, $n_{\mu_i} \equiv \sum_{j,j\neq i} n_{\mu_{ij}}$.

$E_{ij} \triangleq$ Mnemonic for "evaluate." Mapping to the inputs required for A_i from the compressed μ_{ij} from D_j. For example, $M_{ij} = E_{ij}(X_D, \mu_{ij})$ could be the evaluator of a spline approximating structural loads with coefficients μ_{ij}, or if μ_{ij} is the vector of coefficients of a spline fit to pressure, then the convention is that E_{ij} also performs the integration to obtain loads. We will assume that for each F_{ij} there is a corresponding E_{ij}; some of the E_{ij} may be identity mappings and some may be zero maps.

The reader will see that the separation between A_i and its evaluators and the flow of information only in the direction from the evaluators to the analysis code are likely simplifications of the true relationships of these components. For instance, if A_i is a code involving an adaptive grid, in the course of performing its analysis A_i may need to return to its evaluators to obtain information for the adaptively updated grid. This complication is not a problem if the reader bears in mind that the purpose here is to represent the flow of information between disciplines.

Because of the expense of executing the analysis codes, A_i might be used to represent a driver routine that uses some input parameter to decide which of a suite of solver codes with varying fidelity to call at this stage of the design optimization. That is an implementation question, but the ability to handle this situation shows some of the power of the abstraction.

For all of the preceding, A, G, F, E, and W will denote the long vector functions comprising all of the corresponding subscripted functions, for all i, j. To use this compact notation, it is necessary to keep in mind that the ordering of the components must be different to be consistent. For example, suppose that D_1 is aerodynamics and D_2 is structures. Then if we order U as U_1, U_2, we must order A as A_1, A_2 and we must order G as G_{12}, G_{21} to have the convenience of writing $U = A(X_D, G(X_D, U))$ to express the equilibrium of the aeroelastic system.

3.3. Optimization variables.

$X_D \triangleq$ Original problem design variables. These could include wing-shape parameters, beam thicknesses, etc. There are n_D original problem design variables.

$X_\mu, X_U, X_M \triangleq$ Optimizer-controlled values, respectively, for μ, U, M. These are not used in all formulations. In some formulations, the optimizer will explicitly control not only X_D, but also a subset of these surrogates for μ, U, M. They look just like design variables to D_i.

$X \triangleq$ The long vector comprising X_D and any of X_μ, X_U, X_M that are explicitly controlled by the optimizer. For convenience, we will sometimes use X as surrogate

arguments in a function that we have defined above in terms of the original system variables. For example, if we have specified in a particular MDO formulation that X has components X_D, X_μ, then we may write $M = E(X)$.

3.4. Optimization objective and constraints.

$f \triangleq$ Design objective function to be minimized. This could be deviation from desired pressure distribution, drag, weight, etc. In general, f depends explicitly on the design variables X_D and the outputs U of all the analysis disciplines.

$C_D \triangleq$ Original problem design constraints. These could include required lift, maximum allowable stresses, maximum wing length, etc. The constraints in the original problem depend explicitly on the design variables X_D and the outputs U of all the analysis disciplines. These constraints may also include constraints that ensure that the output from one discipline is acceptable as input to another discipline.

$C_{\text{aux}} \triangleq$ Coupling constraints among or within the disciplines needed for formulations in which the optimizer explicitly controls more parameters than X_D, and will change with the formulation chosen. These constraints ensure that feasibility for the reformulated MDO problem is achieved at optimization convergence. For example, if the optimizer controls a surrogate X_U for U, then an auxiliary constraint like $C_{\text{aux}}(X) \equiv W(X_D, G(X), X_U) = 0$ would be needed for the MDO problem.

$C \triangleq$ The vector function of residuals of all constraints C_D and C_{aux} to be satisfied by the optimizer.

4. Feasibility. As described in §2, a *multidisciplinary analysis*, or MDA, is achieved when the coupled system is satisfied. We say that an MDA has been completed when the values of all the variables do not change upon execution of all mappings shown in Fig. 1 without regard to order.

MDA can be very costly because solving the coupled problem usually involves repeatedly executing the single discipline analysis codes in an iterative process. One way to avoid some of this cost is not to require feasibility until convergence to optimality. However, there will be approaches in which we will require partial feasibility for some very good reasons. The point of this section is to express the notions of feasibility needed later by using the framework provided in the previous section.

We say that a single discipline analysis has been carried out for a particular D_i when $W_i(X_D, M_i; U_i) = 0$ has been *solved* to yield U_i for the given inputs X_D, M_i. This would probably be done by executing an analysis code A_i for the given input. When $W_i(X_D, M_i, U_i) = 0$ we have *single discipline feasibility* for discipline i. Note that we do not use the semicolon to separate the arguments when W_i is viewed as a system of equations that are satisfied when particular values are specified for the arguments (i.e., there is no contextual distinction between input and output). The design variables X_D are fixed for a multidisciplinary analysis but, of course, they vary for a multidisciplinary optimization.

Likewise, using the residual form, we say that we have *individual discipline feasibility* (IDF) when

$$W(X_D, M, U) = 0 \tag{1}$$

or when U has been computed in explicit form as $U \equiv A(X_D, M)$. Equation (1) states that individual discipline feasibility implies that *every* discipline has single discipline feasibility. We emphasize this subtle definition: individual discipline feasibility means

that each and every discipline is independently feasible. It is a result of the use of surrogates for the interdisciplinary variables that single disciplines may be independently feasible without having multidisciplinary feasibility.

We have *multidisciplinary feasibility*, or MDF, when, in addition to individual disciplinary feasibility, the interdisciplinary variables match. In the residual form this is

$$W(X_D, M, U) = 0 \quad \text{and} \quad M = G(X_D, U). \tag{2}$$

In the nonresidual form it is

$$U = A(X_D, M) \quad \text{and} \quad M = G(X_D, U). \tag{3}$$

We can combine each of the residual and nonresidual forms into an equivalent equation:

$$W(X_D, G(X_D, U), U) = 0 \ \text{ or } \ U = A(X_D, G(X_D, U)) \ . \tag{4}$$

For the aeroelastic example, the residual form of (1) is

$$\begin{aligned} W_1(X_D, M_{12}, U_1) &= 0, \\ W_2(X_D, M_{21}, U_2) &= 0, \end{aligned} \tag{5}$$

and the interdisciplinary constraints are

$$\begin{aligned} M_{12} &= G_{12}(X_D, U_2), \\ M_{21} &= G_{21}(X_D, U_1). \end{aligned} \tag{6}$$

Thus in the residual form of MDF, we simultaneously satisfy (5) and (6). The complete nonresidual form is

$$\begin{aligned} U_1 &= A_1(X_D, M_{12}), \\ U_2 &= A_2(X_D, M_{21}), \\ M_{12} &= G_{12}(X_D, U_2), \\ M_{21} &= G_{21}(X_D, U_1). \end{aligned} \tag{7}$$

The residual form of the combined equation (4) that expresses multidisciplinary feasibility in terms of just the variables X_D and U is

$$\begin{aligned} W_1(X_D, G_{12}(X_D, U_2), U_1) &= 0, \\ W_2(X_D, G_{21}(X_D, U_1), U_2) &= 0. \end{aligned} \tag{8}$$

To reiterate, the difference between individual discipline feasibility and multidisciplinary feasibility is the matching of interdisciplinary input and output variables to reflect equilibrium. Since traditional single discipline optimization is just optimization under the constraint (1) for one discipline, $M = G(X_D, U)$ is the constraint that distinguishes both MDA and MDO from their single discipline counterparts.

In some applications it may be expedient always to enforce equilibrium between specified pairs of disciplines D_i, D_j. We model this case by coalescing the pair into a single composite discipline. The term "tight coupling" is sometimes used by engineers

to describe this coalescence at the equation level, in which two disciplines D_i and D_j are conjoined to produce a single analysis code that simultaneously solves

$$\begin{aligned} W_i(X_D, M_i; U_i) &= 0, \\ W_j(X_D, M_j; U_j) &= 0, \\ M_{ij} &= G_{ij}(X_D, U_j), \\ M_{ji} &= G_{ji}(X_D, U_i). \end{aligned} \tag{9}$$

5. MDO formulations. Up to this point, we have used our framework for MDO to discuss various kinds of feasibility for the coupled MDA system. The purpose of this section is to widen our discussion to include optimization.

It is important to clearly distinguish between what we mean by a "formulation" and by an "algorithm." By a formulation we mean specifying the objective, the constraints, and the independent or optimization variables. All of our formulations yield nonlinear programming problems, but they have very different attributes that would influence the way the optimization problem would be solved. By an algorithm we mean the specific sequence of steps that would be carried out to solve the resulting nonlinear programming problem.

The key distinguishing feature in the alternative formulations that we present here is the kind of discipline feasibility maintained at every objective function, constraint, or sensitivity evaluation needed during each optimization iteration. In the "multidisciplinary feasible" (MDF) approach, complete multidisciplinary analysis problem feasibility is maintained at every optimization iteration. In the "individual discipline feasible" (IDF) approach, extra independent variables are introduced so that we can choose to maintain only individual discipline feasibility (i.e., single discipline feasibility for all of the disciplines). Interdisciplinary equilibrium constraints are added as optimization constraints to force these extra variables to values that give a full MDA at optimization convergence. In the "all-at-once" (AAO) approach, all of the analysis variables are optimization variables and all of the analysis discipline equations are optimization constraints. Thus, feasibility in AAO and IDF is guaranteed only at optimization convergence. (We could refer to "all-at-once" as "no discipline feasible," but we feel that "all-at-once" better describes the formulation.) In all formulations, the set of optimization variables includes the design variables. None of these formulations imposes any requirements on the design constraints until optimization convergence.

In the following subsections, first we give the general mathematical specification of the MDO problem formulations and then we give the specialized aeroelastic formulations. The Appendix, §8, illustrates the MDF, IDF, and AAO approaches for a very simple example.

5.1. Formulations for general MDO problems.

5.1.1. Multidisciplinary feasible (MDF) formulation. The most common way of posing MDO problems is, in our terminology, the mulitdisciplinary feasible, or MDF, formulation. In this formulation, the vector of design variables X_D is provided by the optimizer to the coupled system of analysis disciplines and a complete MDA is performed with that value of X_D to obtain the system output variable $U(X_D)$ that is used in evaluating $f(X_D, U(X_D))$ and $C_D(X_D, U(X_D))$. The MDF formulation is

$$\begin{aligned} &\text{minimize} && f(X_D, U(X_D)) \\ &\text{with respect to} && X_D, \\ &\text{subject to} && C_D(X_D, U(X_D)) \geq 0, \end{aligned} \tag{10}$$

where $U(X_D) = A(X_D, G(X_D, U(X_D)))$.

Using traditional terminology derived from linear programming, (see [6, p. 253] or [17]), the reader may recognize MDF as a reduced-basis formulation in which X_D is the nonbasic vector and everything else is a basic vector. Since we will use this concept in a nested way, to avoid possible conflicts with the normal usage of the terms basic and nonbasic, we will say that X_D is an *explicit* variable and that U and all the other variables that arise in the MDA part of the problem are *implicit.*

The reader will see that if a derivative-based method is to be used to solve (10), then a complete MDA is necessary not just at every iteration, but at every point where f or C_D or the derivatives are to be evaluated. This can be very expensive, and finite differences would be especially expensive and tricky because iterative methods for determining U may need to be converged to an accuracy well beyond that required for engineering analysis.

5.1.2. The most general formulation. For the sake of completeness, we state here the most general formulation of the MDO problem. This formulation is only for motivation and is unlikely to be useful for MDO. We state the residual and nonresidual forms together. In this kitchen sink formulation, the optimization variables are $X = (X_D, X_M, X_\mu, X_U)$ and all the conditions for a full MDA are included as auxiliary constraints:

$$
\begin{aligned}
&\text{minimize } f(X_D, X_U) \\
&\text{with respect to } X_D, X_M, X_\mu, X_U, \\
&\text{subject to } C_D(X_D, X_U) \geq 0, \\
&\text{and } X_M - E(X_D, X_\mu) = 0 \text{ and } X_\mu - F(X_D, X_U) = 0\ , \\
&\text{and either } W(X_D, X_M, X_U) = 0 \text{ or } X_U - A(X_D, X_M) = 0\ .
\end{aligned} \tag{11}
$$

5.1.3. All-at-once (AAO) formulation. Now we will consider some interesting formulations between the two extremes of two preceding MDO formulations. The first we call the all-at-once (AAO) approach. In AAO, we do not seek to obtain feasibility for the analysis problem in any sense (individual discipline, multidisciplinary, or even for single equations within a discipline) until optimization convergence is reached. In a way, the optimizer does not "waste" time trying to achieve feasibility when far from an optimum. We take as explicit variables $X = (X_D, X_U)$ and write the formulation in terms of the implicit variable $M(X)$ with the interdisciplinary mapping G as its defining relation.

$$
\begin{aligned}
&\text{minimize} \quad f(X) \quad \text{with respect to } X = (X_D, X_U) \\
&\text{subject to} \quad C_D(X) \geq 0, \\
&\qquad\qquad C_{\text{aux}}(X) \triangleq W(X_D, M(X), X_U) = 0,
\end{aligned} \tag{12}
$$

where $M(X) = G(X)$.

The drawback to (12) is that for practical problems it will generally involve a very large number of constraints (the discrete equations from all of the analysis disciplines) and an even larger number, $n_D + \sum_i n_{U_i}$, of optimization variables. Additionally, some of the constraints may not be very smooth. In AAO the analysis "code" performs a particularly simple function; it evaluates the *residuals* of the analysis equations, rather than *solving* some set of equations. Ultimately, of course, the optimization method for AAO must solve the analysis discipline equations W to attain feasibility. Generally, this means that the solution method must contain all of the special techniques that every single discipline analysis solver contains. It is unlikely that

"equality constraint satisfaction schemes" (e.g., Newton's method) present in existing, general purpose optimization codes would be equal to this task in the case where the constraints represent extremely nonlinear partial differential equations (PDEs), as in aerodynamics.

5.1.4. Individual discipline feasible (IDF) formulation. Another way to avoid a complete MDA every time an objective function, constraint, or sensitivity evaluation is needed is to use an IDF formulation like (13). IDF occupies an "in-between" position on a spectrum where the AAO and MDF formulations represent extremes: for AAO, no feasibility is enforced at each optimization iteration, whereas for MDF, complete multidisciplinary feasibility is required. Between these extremes lie other possibilities that amount to specific decompositions of the work between analysis codes and the optimizer. One such possibility, the IDF approach, maintains individual discipline feasibility, while allowing the optimizer to drive the individual disciplines toward multidisciplinary feasibility and optimality by controlling the interdisciplinary data.

Note that, in this approach, analysis variables have been "promoted" to become optimization variables; in fact, they are indistinguishable from design variables from the point of view of a single analysis discipline solver. In IDF, the specific analysis variables that have been promoted are those that represent communication, or coupling, between analysis disciplines via interdisciplinary mappings. The rest of this section describes IDF methods.

The next formulation is the first instance of an IDF approach. The relation that defines the implicit variable $U(X)$ is just the nonresidual form of (1). Thus, each individual discipline is feasible at every optimization iteration. In this method, M is replaced by an explicit surrogate X_M and the interdisciplinary mapping becomes an auxiliary constraint. The explicit variables are $X = (X_D, X_M)$.

$$
\begin{array}{ll}
\text{minimize} & f(X_D, U(X)) \\
\text{with respect to} & X = (X_D, X_M), \\
\text{subject to} & C_D(X_D, U(X)) \geq 0, \\
 & C_{\text{aux}}(X) \triangleq X_M - G(X_D, U(X)) = 0,
\end{array} \tag{13}
$$

where $U(X) = A(X)$. There are $n_D + \sum_i n_{M_i}$ optimization variables in this "uncompressed" IDF approach.

Notice that an evaluation of $U(X) = A(X)$ involves executing all the single discipline analysis codes with simultaneously available multidisciplinary data X. Therefore, these very expensive computations can be done independently and communication costs are likely to be negligible in comparison. Furthermore, the analysis codes vary widely in the types of computations to be done and will generally be suitable for different hardware environments. Thus a heterogeneous network of computers may be particularly well suited for this formulation.

The drawback to the particular IDF method (13) is the large number of optimization variables. As mentioned earlier, we can take advantage of the data compression $\mu_{ij} = F_{ij}(X_D, U_j)$ and elevate μ rather than M to be explicit variables:

$$
\begin{array}{ll}
\text{minimize} & f(X_D, U(X)) \\
\text{with respect to} & X = (X_D, X_\mu), \\
\text{subject to} & C_D(X_D, U(X)) \geq 0, \\
 & C_{\text{aux}}(X) \triangleq X_\mu - F(X_D, U(X))) = 0,
\end{array} \tag{14}
$$

where $U(X) = A(X_D, E(X))$. Thus, the advantage of this "compressed" or "low-bandwidth" IDF formulation is that the optimizer controls possibly the fewest explicit variables of any IDF formulation, namely, $n_D + \sum_i n_{\mu_i}$.

It is possible to write many more permutations, but we will introduce only one more, the possibility of sequencing the individual disciplines.

5.1.5. Sequenced IDF formulations. In the IDF formulations presented above, the interdisciplinary mapping (coupling) variables sent to each discipline from the other disciplines were made optimization variables and associated auxiliary constraints were imposed. We can create IDF formulations where only some of the coupling variables are optimization variables and the remainder are the actual computed analysis values. For example, consider a two-discipline problem such as the aeroelastic example. The computations in the above IDF method could be sequenced such that one of the analyses is completed prior to starting the other one. Since the inputs to the second analysis would then be available, there would be no need for the optimization variables and constraints for the associated interdisciplinary mapping variables from the first to the second discipline. The usefulness of such a "sequenced IDF" formulation depends on factors such as the difficulty in satisfying the coupling constraints, the cost of computing derivatives for the coupling constraints, the relative behavior of the optimization objective and constraint functions for the two formulations, and the lost opportunity for parallelism by imposing a specified sequence on the analyses.

Many different IDF formulations can be developed by using the option to sequence the individual codes. We interpret the formulation represented by (12) in [25] as a sequenced IDF formulation, and so this is not unique to the present work.

5.1.6. Feasible point formulations vs. feasible point algorithms. It is important to note that the use of some optimization algorithms can blur the distinction between MDO formulations. For example, an optimization algorithm that ensures constraint feasibility at each iteration could reduce, or even eliminate, the distinctions between the formulations presented here.

We examine this issue further by briefly discussing the distinction between a generalized reduced gradient (GRG) approach (see [6, p. 221]), which corresponds to MDF, and an approach applied to a full-space problem like AAO that restores feasibility at every iteration.

Some nonlinear programming algorithms approach optimality along a feasible path by following each optimization step with a step to restore feasibility. This is very different from a generalized reduced gradient approach which maintains feasibility not just for each iterate, but for any pair X_D, U that ever appears in any context in the algorithm. In other words, the GRG approach eliminates U from the optimization calculations by using the implicitly defined function $U(X_D)$ in its stead.

If we apply a feasibility restoration method to an AAO formulation, then at each optimization iteration the optimization algorithm would first take a step in the full space to obtain a complete new X. This would be followed by a so-called restoration step which would consist here of an MDA to replace the X_U part of the AAO optimization iterate X_D, X_U with $U(X_D)$. The next optimization iteration is started from the multidisciplinary feasible point $X = (X_D, U(X_D))$ satisfying (4).

However, such an approach to the AAO formulation is not the same as MDF because X_U is treated as independent of X_D for the purpose of setting the new iterate's X_D. In the MDF or GRG approach, X_D is the only variable in the optimization

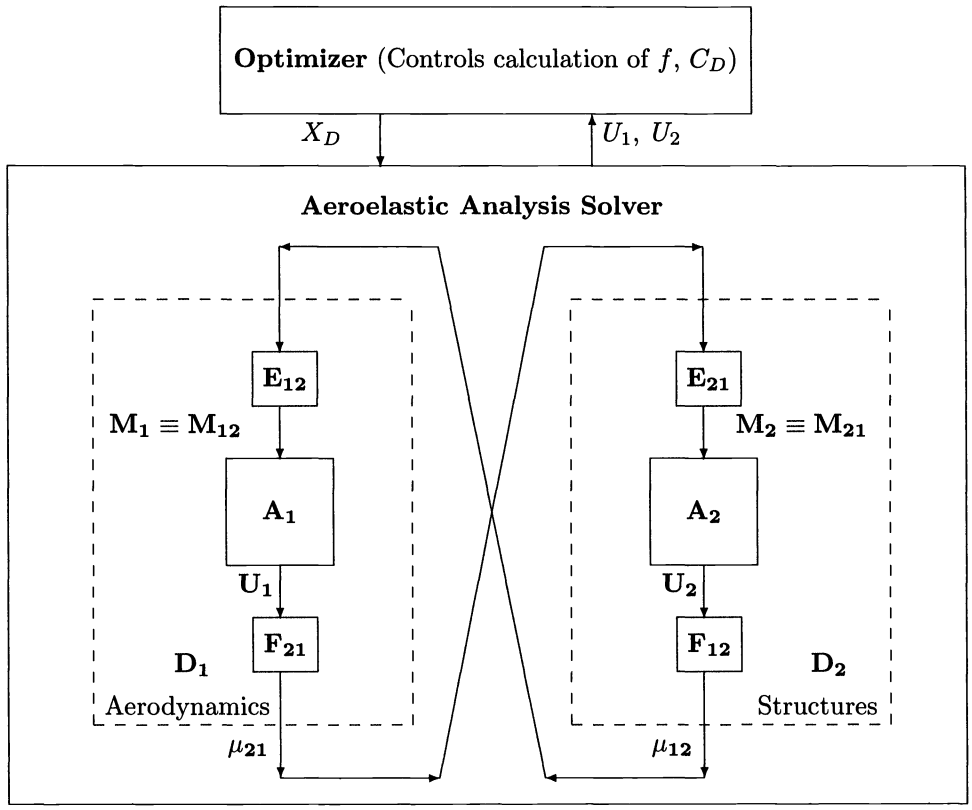


FIG. 3. *Multidisciplinary feasible* (MDF) *method.*

iteration. This means that the derivatives or sensitivities required in the MDF formulation must be computed with arguments X_D and values of all the system variables that correspond to an MDA solution for that value of X_D.

5.2. Formulations specialized to the aeroelastic MDO problem. To further elucidate the formulation ideas and to prepare for a discussion of the sensitivities needed to apply most nonlinear programming (NLP) algorithms to each formulation, we show how the general formulations apply to the specific case of aeroelastic MDO.

First is the standard MDF formulation (10)

$$
\begin{aligned}
&\text{minimize} && f(X_D, U_1(X_D), U_2(X_D)) \quad \text{with respect to } X_D,\\
&\text{subject to} && C_D(X_D, U_1(X_D), U_2(X_D)) \geq 0,\\
&\text{where} && U_1(X_D) = A_1(X_D, G_{12}(X_D, U_2(X_D))),\\
&&& U_2(X_D) = A_2(X_D, G_{21}(X_D, U_1(X_D))).
\end{aligned} \tag{15}
$$

Figure 3 illustrates the MDF formulation. Notice that while Fig. 3 provides some detail about the computations of U_2 and U_1, the aeroelastic analysis is a "black box" from the perspective of the optimization code.

If analysis residuals are available, then one might try to avoid so many costly MDA computations by an all-at-once, or AAO, formulation with U_1 and U_2 made explicit.

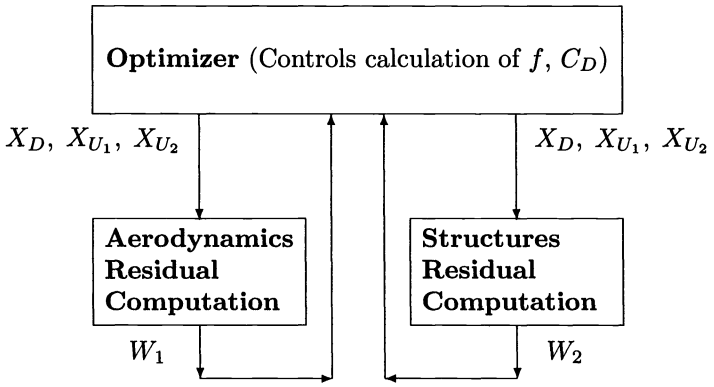


FIG. 4. *All-at-once* (AAO) *method.*

Figure 4 illustrates the the AAO formulation. The AAO optimization problem is

$$
\begin{array}{lll}
 & \text{minimize} & f(X) \\
 & \text{with respect to} & X = (X_D, X_{U_1}, X_{U_2}), \\
(16) & \text{subject to} & C_D(X) \geq 0, \\
 & & W_1(X_D, M_{12}(X), X_{U_1}) = 0, \\
 & & W_2(X_D, M_{21}(X), X_{U_2}) = 0,
\end{array}
$$

where $M_{12}(X) = G_{12}(X_D, X_{U_2})$ and $M_{21}(X) = G_{21}(X_D, X_{U_1})$.

Other all-at-once (AAO) formulations for design optimization problems have been mentioned in the literature for aerodynamic optimization (e.g., [5], [10], [14], [24]), structural optimization (e.g., [7]), chemical process control, and control and inverse problems (e.g., [18], [23], [20]). In [14] this approach is called the "one-shot" method, and in [7] it is called "simultaneous analysis and design." In [5] the authors discuss how AAO can be remarkably efficient for aerodynamic optimization, provided some computational difficulties can be overcome.

The rest of this section describes two aeroelastic IDF methods. We reiterate the essence of IDF: at each optimization iteration we have a "correct" aerodynamic analysis and a "correct" structural analysis; however, it is only at optimization convergence that the pressures predicted by the aerodynamic analysis correspond to the loads sent to the structures and the displacements predicted by the structural analysis correspond to the geometry sent to the aerodynamics code. Again, we remind the reader that one could follow each optimization step by performing a feasibility restoring MDA, but the optimization problem being solved would still be an IDF and not an MDF formulation.

The "uncompressed" IDF formulation is

$$
\begin{array}{lll}
 & \text{minimize} & f(X_D, U_1(X), U_2(X)) \\
 & \text{with respect to} & X = (X_D, X_{M_{12}}, X_{M_{21}}), \\
(17) & \text{subject to} & C_D(X_D, U_1(X), U_2(X)) \geq 0, \\
 & & C_{12} \equiv X_{M_{12}} - G_{12}(X_D, U_2(X)) = 0, \\
 & & C_{21} \equiv X_{M_{21}} - G_{21}(X_D, U_1(X)) = 0,
\end{array}
$$

where $U_1(X) = A_1(X_D, X_{M_{12}})$ and $U_2(X) = A_2(X_D, X_{M_{21}})$.

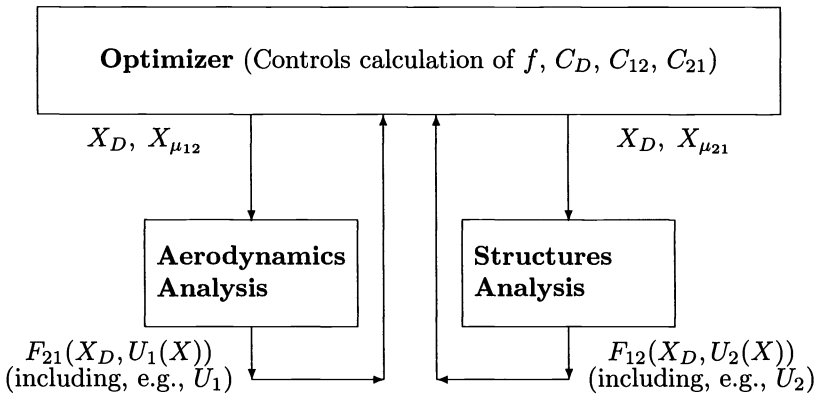


FIG. 5. *Low-bandwidth individual discipline feasible* (IDF) *method.*

The low-bandwidth IDF formulation is

$$
\begin{array}{lll}
 & \text{minimize} & f(X_D, U_1(X), U_2(X)) \\
 & \text{with respect to} & X = (X_D, X_{\mu_{12}}, X_{\mu_{21}}), \\
(18) & \text{subject to} & C_D(X_D, U_1(X), U_2(X)) \geq 0, \\
 & & C_{12} \equiv X_{\mu_{12}} - F_{12}(X_D, U_2(X)) = 0, \\
 & & C_{21} \equiv X_{\mu_{21}} - F_{21}(X_D, U_1(X)) = 0,
\end{array}
$$

where $U_1(X) = A_1(X_D, E_{12}(X_D, X_{\mu_{12}}))$ and $U_2(X) = A_2(X_D, E_{21}(X_D, X_{\mu_{21}}))$. Figure 5 shows the flow of information for this low-bandwidth IDF formulation.

6. Derivative requirements for MDO. We anticipate that most MDO efforts will involve derivative-based optimization algorithms. For this reason we now discuss the derivatives required in the MDF, IDF, and AAO formulations that we have presented. If one looks in the previous section, all the formulations given either use the analysis residuals as a constraint, or else the analysis code solution mapping is used to define U as an implicit variable. Thus, any derivative-based algorithm will require either the derivative of the analysis residuals or of the solution operator.

As mentioned above, AAO has the disadvantage that the optimization code must assume the difficult task of simultaneously satisfying all the analysis discipline equations. The MDF and IDF formulations have the advantage that they use the specialized software A_i that has been developed for solving the individual discipline equations. But there is a price to be paid for using the existing software; the MDF and IDF formulations must differentiate the solution operators implemented by the single discipline solvers.

The most daunting task is to obtain these solution sensitivities. Perhaps the most obvious approach is to use finite difference approximations [16]. This certainly finesses the issue, but because of problems with accuracy and expense, we believe that a practical alternative to finite differences must be found if MDO is to become an everyday engineering tool. There seem to be two alternatives: analytic approaches (implicit differentiation, sensitivity equations, adjoint equation solution) and automatic differentiation.

Even though there is considerable research interest in analytic methods for sensitivity or gradient calculations [1], [2], [7]–[9], [11], [12], few analysis codes in engineering use today provide the required derivatives. We hope that automatic differentiation will provide tools that will help us retrofit existing codes to produce the derivatives.

Automatic differentiation should not be confused with symbolic differentiation. In ADIFOR [19], the automatic differentiation tool with which we are the most familiar, the definition of the function is given as a standard Fortran program. The output from ADIFOR is a program that duplicates the computation of the original program, and, in addition, it includes code to compute the sensitivities of indicated outputs with respect to indicated outputs. The sensitivities are computed with the same accuracy as the quantities whose partial derivatives they represent.

We finish our discussion of the problems of finding derivatives with a brief discussion of the derivatives needed by the various formulations. There seems to be little point to laboring through the general case, and so we will restrict ourselves to the aeroelastic example problem. In the subsections that follow we give the derivatives of the constraints. However, the objective function f and the design constraints C_D depend on the same parameters. Thus, the gradient of the objective function is just the transpose of the block row of the constraint Jacobian corresponding to C_D, with C_D replaced by f.

6.1. Derivatives required for the MDF formulation. For MDF optimization, Sobieszczanski-Sobieski [13] gives a complete presentation of the alternative approaches, but our MDO model includes the fit and evaluate routines and so the form is slightly different here.

Using the MDF formulation given by (15), the linearized constraint residual is, noting $X = [X_D]$,

$$\left[C_D^{(c)} \right] + \left[\frac{\partial C_D}{\partial X_D} + \frac{\partial C_D}{\partial U_1}\frac{\partial U_1}{\partial X_D} + \frac{\partial C_D}{\partial U_2}\frac{\partial U_2}{\partial X_D} \right] \left[\Delta X_D \right], \tag{19}$$

where $C_D^{(c)}$ is the value of the constraints at the current approximation $X_D^{(c)}$ to the solution of the MDO problem. Remember, for the MDF formulation, $X_D^{(c)}$ is a full MDA solution point. The coefficient matrix in (19) represents the Jacobian of C_D with respect to the design variables X_D.

Computing the partial derivatives $\partial f/\partial X_D$, $\partial f/\partial U_\alpha$, $\partial C_D/\partial X_D$, and $\partial C_D/\partial U_\alpha$ for $\alpha = 1, 2$ is generally easy; computing the solution sensitivities $\partial U_\alpha/\partial X_D$ is generally hard. One way to obtain these sensitivities is to form and solve a linear system as follows. Notice that since we need the derivatives at a point $X_D^{(c)}$ for which (7) holds, we can apply implicit differentiation to the appropriate residual equations to obtain:

$$\frac{\partial W_1(X_D, M_{12}, U_1)}{\partial X_D} + \frac{\partial W_1(X_D, M_{12}, U_1)}{\partial M_{12}} \left[\frac{\partial G_{12}}{\partial X_D} + \frac{\partial G_{12}}{\partial U_2}\frac{\partial U_2}{\partial X_D} \right] + \frac{\partial W_1(X_D, M_{12}, U_1)}{\partial U_1}\frac{\partial U_1}{\partial X_D} = 0$$

and

$$\frac{\partial W_2(X_D, M_{21}, U_2)}{\partial X_D} + \frac{\partial W_2(X_D, M_{21}, U_2)}{\partial M_{21}} \left[\frac{\partial G_{21}}{\partial X_D} + \frac{\partial G_{21}}{\partial U_1}\frac{\partial U_1}{\partial X_D} \right] + \frac{\partial W_2(X_D, M_{21}, U_2)}{\partial U_2}\frac{\partial U_2}{\partial X_D} = 0, \tag{20}$$

where we have used the fact that $M = G(X_D, U)$. Every derivative in (20) is assumed to be available except the derivatives of U. Thus, we can gather terms to obtain a

linear system that can be solved to obtain the sought for U derivatives:

$$\begin{bmatrix} \frac{\partial W_1}{\partial U_1} & \frac{\partial W_1}{\partial M_{12}}\frac{\partial G_{12}}{\partial U_2} \\ \frac{\partial W_2}{\partial M_{21}}\frac{\partial G_{21}}{\partial U_1} & \frac{\partial W_2}{\partial U_2} \end{bmatrix} \begin{bmatrix} \frac{\partial U_1}{\partial X_D} \\ \frac{\partial U_2}{\partial X_D} \end{bmatrix} = - \begin{bmatrix} \frac{\partial W_1}{\partial X_D} + \frac{\partial W_1}{\partial M_{12}}\frac{\partial G_{12}}{\partial X_D} \\ \frac{\partial W_2}{\partial X_D} + \frac{\partial W_2}{\partial M_{21}}\frac{\partial G_{21}}{\partial X_D} \end{bmatrix}. \tag{21}$$

If the residuals are not available, then in a similar way, we can differentiate (7) and rearrange terms to obtain

$$\begin{bmatrix} I & -\frac{\partial U_1}{\partial M_{12}}\frac{\partial G_{12}}{\partial U_2} \\ -\frac{\partial U_2}{\partial M_{21}}\frac{\partial G_{21}}{\partial U_1} & I \end{bmatrix} \begin{bmatrix} \frac{\partial U_1}{\partial X_D} \\ \frac{\partial U_2}{\partial X_D} \end{bmatrix} = \begin{bmatrix} \frac{\partial U_1}{\partial X_D} + \frac{\partial U_1}{\partial M_{12}}\frac{\partial G_{12}}{\partial X_D} \\ \frac{\partial U_2}{\partial X_D} + \frac{\partial U_2}{\partial M_{21}}\frac{\partial G_{21}}{\partial X_D} \end{bmatrix}, \tag{22}$$

where we have used (7) to replace partial derivatives of A_1 and A_2 by partial derivatives of U_1 and U_2, respectively. It is important to remember here that *all* of these partials must be evaluated at $X_D^{(c)}$ and $U(X_D^{(c)})$, a multidisciplinary feasible point.

6.2. Derivatives required for the AAO formulation. The AAO formulation has much easier derivative requirements because the derivatives do not have to be evaluated at MDA solutions. In fact, for formulation (12), there is no feasibility required at arbitrary evaluation points. The linearized constraint residual is, noting $X = [X_D, X_U]$ and $X_U = [X_{U_1}, X_{U_2}]$,

$$\begin{bmatrix} C_D^{(c)} \\ W_1^{(c)} \\ W_2^{(c)} \end{bmatrix} + \begin{bmatrix} \frac{\partial C_D}{\partial X_D} & \frac{\partial C_D}{\partial X_{U_1}} & \frac{\partial C_D}{\partial X_{U_2}} \\ \frac{\partial W_1}{\partial X_D} & \frac{\partial W_1}{\partial X_{U_1}} & \frac{\partial W_1}{\partial X_{U_2}} \\ \frac{\partial W_2}{\partial X_D} & \frac{\partial W_2}{\partial X_{U_1}} & \frac{\partial W_2}{\partial X_{U_2}} \end{bmatrix} \begin{bmatrix} \Delta X_D \\ \Delta X_{U_1} \\ \Delta X_{U_2} \end{bmatrix}, \tag{23}$$

where $(C_D^{(c)}, W_1^{(c)}, W_2^{(c)})^T$ is the residual of the constraints at the current point.

The blocks $\partial W_1/\partial X_{U_1}$ and $\partial W_2/\partial X_{U_2}$ in (23) are the Jacobians that would appear in Newton solvers for the two disciplines, respectively. The derivatives $\partial W_1/\partial X_D$, $\partial W_1/\partial X_{U_2}$, $\partial W_2/\partial X_D$, and $\partial W_2/\partial X_{U_1}$ represent the sensitivities of the analysis discipline equation residuals to their inputs from other disciplines. We assume that these derivatives are available.

6.3. Derivatives required for the IDF formulations. The linearized constraint residual for the IDF formulation (17) is

$$\begin{bmatrix} C_D^{(c)} \\ C_{12}^{(c)} \\ C_{21}^{(c)} \end{bmatrix} + J \begin{bmatrix} \Delta X_D \\ \Delta X_{M_{12}} \\ \Delta X_{M_{21}} \end{bmatrix}, \tag{24}$$

where

$$J = \begin{bmatrix} \frac{dC_D}{dX_D} & \frac{\partial C_D}{\partial U_1}\frac{\partial U_1}{\partial X_{M_{12}}} & \frac{\partial C_D}{\partial U_2}\frac{\partial U_2}{\partial X_{M_{21}}} \\ -\frac{\partial G_{12}}{\partial X_D} - \frac{\partial G_{12}}{\partial U_2}\frac{\partial U_2}{\partial X_D} & I & -\frac{\partial G_{12}}{\partial U_2}\frac{\partial U_2}{\partial X_{M_{21}}} \\ -\frac{\partial G_{21}}{\partial X_D} - \frac{\partial G_{21}}{\partial U_1}\frac{\partial U_1}{\partial X_D} & -\frac{\partial G_{21}}{\partial U_1}\frac{\partial U_1}{\partial X_{M_{12}}} & I \end{bmatrix}, \tag{25}$$

TABLE 1
Comparison of formulation features.

	All-at-once (AAO)	(Compressed) individual discipline feasible (IDF)	Multidisciplinary feasible (MDF)
Use of existing analysis codes	None	Full, no direct coupling of analysis codes	Full, but must couple the analysis codes
Discipline feasibility	None until optimal, then all disciplines feasible	Individual discipline feasibility at each optimization iteration	Multidisciplinary feasibility at each optimization iteration
Variables the optimizer controls. (Thus, these are independent variables in sensitivities.)	Design variables and all analysis discipline unknowns	Design variables and interdisciplinary mapping (coupling) parameters	Design variables
Number of optimization variables. (Thus, the number of sensitivities required.)	$n_D + \sum_i n_{U_i}$	$n_D + \sum_i n_{\mu_i}$	n_D
Optimization problem size and sparsity	Very large and sparse	Moderate, size and sparsity dependent on coupling "bandwidth"	Small and dense

$$\frac{dC_D}{dX_D} = \frac{\partial C_D}{\partial X_D} + \frac{\partial C_D}{\partial U_1}\frac{\partial U_1}{\partial X_D} + \frac{\partial C_D}{\partial U_2}\frac{\partial U_2}{\partial X_D}, \tag{26}$$

and $(C_D^{(c)}, C_{12}^{(c)}, C_{21}^{(c)})^T$ is the value of the constraints at the current point. Derivatives similar to the above can be derived for the low-bandwidth IDF formulation (18).

The expensive derivatives for the IDF method are those of the form $\partial U_i/\partial X_D$, $\partial U_i/\partial X_{M_{ij}}$, and $\partial U_i/\partial X_{\mu_{ij}}$, which are all sensitivities of the individual discipline analysis solutions with respect to either uncompressed or compressed analysis inputs. Note that the derivatives required for the IDF formulation are the same as those required in (22) and by Sobieszczanski-Sobieski [13] (in his GSE2 approach) for computing MDF problem derivatives. However, in contrast to the MDF method, here they only need to be *evaluated at an* individual discipline feasibility *point.*

7. Concluding remarks. In Table 1 we compare the features of our three main approaches to MDO formulation. In Table 2 we speculate on the performance that might be achieved by the approaches. These hypotheses are supported by the experimental results shown in [26] and [21].

The multidisciplinary feasible (MDF) and individual discipline feasible (IDF) approaches have the advantage of using, with moderate or no modification, existing single discipline analysis codes. An additional advantage of IDF is that it avoids

TABLE 2
Comparison of predicted performance.

	All-at-once (AAO)	Individual discipline feasible (IDF)	Multidisciplinary feasible (MDF)
Probable compute time for objective and constraints	Low, evaluate residuals for all disciplines	Moderate, separately analyze each discipline	Very high, full multidisciplinary analysis
Expected overall speed of optimization process	Fast	Medium	Slow
Probability of unanalyzable intermediate designs	Low	Medium	High
Probable robustness	Unknown	High	Medium

the cost of achieving full multidisciplinary feasibilty at each optimization iteration, a procedure that is probably wasteful in MDF when far from optimization convergence. Furthermore, the IDF method makes it easy to replace one analysis code with another (as when additional modeling fidelity is required), or to add new disciplines.

On the other hand, the IDF approach requires the explicit imposition in the optimization of the nonlinear constraints involving the interdisciplinary maps and the calculation of additional sensitivities corresponding to the variables communicated between disciplines. If the number of such variables and constraints can be kept small, we project that the overall cost of IDF optimization will be significantly less than MDF optimization.

We feel that the all-at-once (AAO) approach remains theoretically attractive because of the probability that it will be the least expensive computationally. Unfortunately, it requires a higher degree of software integration than is likely to be achieved in the near future for realistic applications.

No matter what approach is chosen, the efficient calculation of sensitivities will be critical for success. In our opinion, with the increasing complexity of analysis codes and the increasing number of design variables that will probably be used in future MDO applications, it is unlikely that finite difference sensitivities will be affordable. In this area, the role of automatic differentiation remains to be conclusively determined. Our guess is that, for very large problems, only some kind of analytic or implicit sensitivities will be used. The other alternative, of course, is to use simplified analyses in the optimization and then to correct via iterative refinement. For example, this approach has been used in multidisciplinary design of helicopter rotors [15]. We observe that this approach dovetails well with the IDF approach, where an existing multidisciplinary analysis procedure can be viewed as "one discipline," and information of higher fidelity for a single analysis code can be viewed as the "second discipline" [22]. The iterative refinement outer loop could build a simplified model of the higher fidelity code. The iterative refinement optimization loop would then use the IDF method with the multidisciplinary analysis as one discipline and the simplified model as the second discipline.

MDO problems will often involve design over several cases, or design points. In

the aeroelastic problem, for example, there may be stress constraints for several flight conditions such as pull-up or dive maneuvers. There may also be minimum performance requirements for off-design values of velocity, altitude, etc. In fact, different analysis codes may be used for different design points. For example, a low-fidelity aerodynamics analysis code may be acceptable for computing pressures for a dive maneuver, while a high-fidelity aerodynamics analysis code may be required for computing drag and lift at cruise. MDO over multiple design points can be readily couched in the formulations presented here by considering the analyses at each point to be a separate "discipline."

Because of the large size and computational cost of solving MDO problems, use of parallel and distributed computing will probably be required. Clearly, the IDF formulation approach is particularly well suited to implementation in a heterogeneous computing environment consisting of computers that are separately suitable for each individual disciplinary analysis. Additionally, both the MDF and IDF approaches, since they leave intact the disciplinary analysis codes, maintain all of the parallelism that might have been developed by disciplinary specialists within a single discipline's solver.

In this paper we did not concentrate on optimization solution methods for MDO. Although Frank et al. [4] surveyed various optimization techniques that could be applied to MDO problems, to our knowledge no one has systematically investigated the application of specific mathematical programming algorithms to various MDO formulations.

8. Appendix. Simple example. Here we illustrate the fundamental aspects of MDF, AAO, and IDF by applying these formulations to a simple example. It is important to note that this example does not exhibit many important characteristics, like interdisciplinary mappings, inherent in realistic MDO problems.

Let U_1 and U_2 be scalar analysis variables and let X_D be a scalar design variable. Consider two "disciplines" defined by

$$W_1(X_D, U_2, U_1) = U_1 + \sin U_2 - U_2 \sin U_1 + X_D U_2 = 0, \tag{27}$$

$$W_2(X_D, U_1, U_2) = U_2 + U_1 \cos U_2 + \cos U_1 + X_D U_1 = 0, \tag{28}$$

with the objective to minimize $f = U_1^2 + U_2^2$. When viewed as single disciplines, (27) is solved for $U_1 = A_1(X_D, U_2)$ and (28) is solved for $U_2 = A_2(X_D, U_1)$. When viewed as a multidisciplinary system, X_D is given and (27) and (28) are simultaneously solved for U_1 and U_2. The three formulations for this example are given below.

8.1. MDF. The MDF formulation is

$$\begin{array}{ll} \text{minimize} & f = [U_1(X_D)]^2 + [U_2(X_D)]^2 \\ \text{with respect to} & X_D. \end{array} \tag{29}$$

Here, for any X_D, f is computed by solving (27) and (28) simultaneously for U_1 and U_2.

8.2. AAO. Let X_{U_1} and X_{U_2} be surrogates for U_1 and U_2, respectively. The AAO formulation is

$$\begin{array}{ll} \text{minimize} & f = [X_{U_1}]^2 + [X_{U_2}]^2 \\ \text{with respect to} & X_D, X_{U_1}, X_{U_2}, \\ \text{such that} & X_{U_1} + \sin X_{U_2} - X_{U_2} \sin X_{U_1} + X_D X_{U_2} = 0, \\ & X_{U_2} + X_{U_1} \cos X_{U_2} + \cos X_{U_1} + X_D X_{U_1} = 0. \end{array} \tag{30}$$

8.3. IDF. With the above-defined surrogates, the IDF formulation is

$$
\begin{array}{ll}
\text{minimize} & f = [U_1(X_D, X_{U_2})]^2 + [U_2(X_D, X_{U_1})]^2 \\
\text{with respect to} & X_D, X_{U_1}, X_{U_2}, \\
\text{such that} & X_{U_1} - U_1(X_D, X_{U_2}) = 0, \\
& X_{U_2} - U_2(X_D, X_{U_1}) = 0.
\end{array} \tag{31}
$$

Here, U_1 is computed as follows. Given X_D, and with the value of X_{U_2} substituted for U_2, (27) can be solved for the remaining unknown U_1. U_2 is computed similarly, but independently, from (28). In other words, $U_1 = A_1(X_D, X_{U_2})$ and $U_2 = A_2(X_D, X_{U_1})$.

Acknowledgments. The authors gratefully acknowledge comments made by Jarek Sobieszczanski-Sobieski and by an anonymous referee that helped us improve this paper.

REFERENCES

[1] O. BAYSAL, AND E. M. ELESHAKY, *Aerodynamic design using sensitivity analysis and computational fluid dynamics*, Tech. Report AIAA-91-0471, AIAA, January, 1991, presented at the 29th Aerospace Sciences Meeting, January 7–10, 1991, Reno, NV.

[2] J. BORGGAARD, J. A. BURNS, E. M. CLIFF, AND M. GUNZBURGER, *Sensitivity calculations for 2d, inviscid, supersonic forebody problem*, Tech. Report 93-13, Institute for Computer Applications in Science and Engineering, Hampton, VA, March, 1993.

[3] E. J. CRAMER, P. D. FRANK, G. R. SHUBIN, J. E. DENNIS, JR., AND R. M. LEWIS, *On alternative problem formulations for multidisciplinary design optimization*, AIAA-92-4752, 4th Symposium on Multidisciplinary Analysis and Optimization, September 21–23, 1992, Cleveland, OH.

[4] P. D. FRANK, A. J. BOOKER, T. P. CAUDEL, AND M. J. HEALY, *Optimization and search methods for multidisciplinary design*, AIAA-92-4827, 4th Symposium on Multidisciplinary Analysis and Optimization, September 21–23, 1992, Cleveland, OH.

[5] P. D. FRANK AND G.R. SHUBIN, *A comparison of optimization-based approaches for a model computational aerodynamics design problem*, J. Comput. Physics, 98 (1992), pp. 74–89.

[6] P. E. GILL, W. MURRAY, AND M. H. WRIGHT, *Practical Optimization*, Academic Press, New York, 1981.

[7] R. T. HAFTKA, Z. GURDAL, AND M. P. KAMAT, *Elements of Structural Optimization*, Kluwer Academic Publishers, Norwell, MA, 1990.

[8] A. JAMESON, *Aerodynamic design via control theory*, Tech. Report 88-64, Institute for Computer Applications in Science and Engineering, Hampton, VA, November, 1988.

[9] V. KORIVI, A. TAYLOR, III, P. NEWMAN, AND G. HOU, AND H. JONES, *An approximately factored incremental strategy for calculating consistent discrete CFD sensitivity derivatives*, AIAA-92-4746, 4th Symposium on Multidisciplinary Analysis and Optimization, September 21–23, 1992, Cleveland, OH.

[10] M. H. RIZK, *Aerodynamic optimization by simultaneously updating flow variables and design parameters*, AGARD Paper No. 15, May 1989.

[11] G. R. SHUBIN, *Obtaining "cheap" optimization gradients from computational aerodynamics codes*, Tech. Report AMS-TR-164, Boeing Computer Services, Seattle, WA, June, 1991.

[12] G. R. SHUBIN AND P. D. FRANK, *A comparison of two closely-related approaches to aerodynamic design optimization*, in Proceedings of the Third International Conference on Inverse Design Concepts and Optimization in Engineering Sciences (ICIDES-III), G. S. Dulikravich, ed., October, 1991.

[13] J. SOBIESZCZANSKI-SOBIESKI, *Sensitivity of complex, internally coupled systems*, AIAA J., 28 (1990), pp. 153–160.

[14] S. TA'ASAN, G. KURUVILA, AND M. D. SALAS, *Aerodynamic design and optimization in one shot*, AIAA Paper 92-0025, 30th Aerospace Sciences Meeting, Reno, NV, January, 1992.

[15] D. K. YOUNG AND F. J. TARZANIAN JR., *Structural optimization and Mach scale test validation of a low vibration rotor*, 47th Annual Forum of the American Helicopter Society, Phoenix, AZ, 1991.

[16] AIAA Technical Committee on Multidisciplinary Design Optimization (MDO), White paper on current state of the art, American Institute of Aeronautics and Astronautics, 1991.

[17] M. AVRIEL, *Nonlinear Programming*, Prentice-Hall, New York, 1976.
[18] H. T. BANKS AND K. KUNISCH, *Estimation Techniques for Distributed Parameter Systems*, Birkhauser, 1989.
[19] C. BISCHOF, A. CARLE, G. CORLISS, A. GRIEWANK, AND P. HOVLAND, ADIFOR: *generating derivative codes from Fortran programs*, Tech. Report CRPC-TR91185, Center for Research on Parallel Computation, December, 1991.
[20] C. R. HARGRAVES AND S. W. PARIS, *Direct trajectory optimization using nonlinear programming and collocation*, J. Guidance, Control, Dynamics, 10 (1987), p. 338.
[21] J. E. DENNIS, G. LI, AND K. WILLIAMSON, *Optimization algorithms for parameter identification*, Tech. Report CRPC-TR92277, Center for Research on Parallel Computation, January, 1992.
[22] D. L. GROSE, private communication.
[23] F.-S. KUPFER AND E. W. SACHS, *A prospective look at SQP methods for semilinear parabolic control problems*, in Optimal control of partial differential equations: proceedings of the IFIP WG 7.2 International Conference, 1991, pp. 145–157,
[24] C. E. OROZCO AND O. N. GHATTAS, *Massively parallel aerodynamic shape optimization*, preprint.
[25] R. T. HAFTKA, J. SOBIESZCZANSKI-SOBIESKI, AND S. L. PADULA, *On options for interdisciplinary analysis and design optimization*, Structural Optim., 4 (1992), pp. 65–74.
[26] G. R. SHUBIN, *Application of alternative multidisciplinary optimization formulations to a model problem for static aeroelasticity*, Tech. Report BCSTECH-93-022, Boeing Computer Services, Seattle, December, 1993.
[27] P. L. YU, *Multiple criteria decision making: five basic concepts*, in Handbooks in Operations Research and Management Science, G. L. Nemhauser, A. H. G. Rinooy Kan, and M. J. Todd, eds., Elsevier, North-Holland, 1989, pp. 663–699.

SIAM J. OPTIMIZATION
Vol. 4, No. 4, pp. 777–793, November 1994
© 1994 Society for Industrial and Applied Mathematics
006

COORDINATION IN COARSE-GRAINED DECOMPOSITION*

R. DE LEONE†, R. R. MEYER†, S. KONTOGIORGIS†,
A. ZAKARIAN†, AND G. ZAKERI†

Abstract. The dominance of parallel computers in the supercomputer field provides great incentive for the development of parallel algorithms in mathematical programming. Fortunately, parallel computers provide a good medium for the implementation of many types of decomposition algorithms for large-scale optimization. In this paper, a variety of coordination methods are studied for coarse-grained decomposition techniques. Some are variants of existing approaches that are designed to enhance rates of convergence and/or parallel efficiency, while others are novel methods that try to take maximum advantage of the parallel environment.

Key words. large scale optimization, decomposition techniques, parallel computing

AMS subject classifications. 90C06, 90C35, 49M27, 65Y05

1. Introduction.

The overall goal of the research reported here is to shed light on the trade-off between the complexity of the coordinating mechanism used in the decomposition process and the overall performance in a parallel environment. Historically, there has been a basic dichotomy between complex coordinators that requires the solution of complex optimization problems (such as in the Dantzig–Wolfe [1] or Schultz–Meyer [20], [21] methods) and simple coordinators that require only summing and averaging of subproblem information (as in the Douglas–Rachford splitting methods [2]). Methods based on simple coordinators are, on the surface, better suited to parallel environments since they avoid the potential serial bottleneck of solving the possibly large single coordinating problem associated with complex coordinators. Although this bottleneck could be attacked by attempting to solve the coordination problem itself in parallel, it is not always possible to do this efficiently, especially since such problems may be nonlinear and, even when linear, do not have structural properties that allow a natural decomposition. On the other hand, although utilization of simple coordinators takes great advantage of the universal architecture of the Connection Machine CM–5 supercomputer [24] (in which powerful processors may operate independently in data parallel fashion on the subproblems and then efficiently combine and share results via a control network), this coordination approach may lead to a large number of outer iterations (here an outer iteration corresponds to a set of subproblem solutions), and hence slow convergence relative to methods with complex coordinators.

We present computational experience with multicommodity flow problems that demonstrates these trade-offs, and we propose and prove convergence of a new multi-coordinator approach in which each coordination step involves the solution of many "small" coordination problems—a process that is inherently parallel.

* Received by the editors September 14, 1993; accepted for publication (in revised form) December 20, 1993. This research was supported by Air Force Office of Scientific Research grant AFOSR-89-0410 and by National Science Foundation grants CCR-8907671, CDA-9024618, and CCR-9306807.

† Center for Parallel Optimization, Computer Sciences Department, University of Wisconsin–Madison, 1210 West Dayton Street, Madison, Wisconsin 53706 (rrm@cs.wisc.edu).

The problem that we initially consider is the linear block-angular problem:

$$
\begin{aligned}
z = \min_{x_{[1]},x_{[2]},\ldots,x_{[K]}} \quad & {c_{[1]}}^T x_{[1]} + {c_{[2]}}^T x_{[2]} + \cdots \cdots + {c_{[K]}}^T x_{[K]} \\
& A_{[1]} x_{[1]} = b_{[1]}, \\
& \qquad A_{[2]} x_{[2]} = b_{[2]}, \\
& \qquad\qquad \ddots \\
& \qquad\qquad\quad \ddots \\
& \qquad\qquad\qquad A_{[K]} x_{[K]} = b_{[K]} \\
& D_{[1]} x_{[1]} + D_{[2]} x_{[2]} + \cdots \cdots + D_{[K]} x_{[K]} \le d \\
& 0 \le x_{[k]} \le u_{[k]} \qquad k = 1, \ldots, K,
\end{aligned} \tag{1}
$$

where for each $k = 1, \ldots, K$, $A_{[k]}$ and $D_{[k]}$ are, respectively, $m_k \times n_k$ and $m_0 \times n_k$ real matrices and $c_{[k]} \in \mathbb{R}^{n_k}$. Some of the components of $u_{[k]}$ can be $+\infty$. Later on we consider the nonlinear block-angular problem, in which the linear objective in (1) is replaced by

$$f_{[1]}(x_{[1]}) + f_{[2]}(x_{[2]}) + \cdots + f_{[K]}(x_{[K]}),$$

where $f_{[k]}$ is a convex function from $\mathbb{R}^{n_k}$ to $\mathbb{R}$. We assume that the above problem (1) is solvable.

The paper is organized as follows. In §2 we briefly describe the Dantzig–Wolfe method and its parallel implementation on the Connection Machine CM–5. Section 3 presents the Douglas–Rachford decomposition scheme. Multicoordination methods are described in §4. Computational results on the Connection Machine CM–5 for the Dantzig–Wolfe method and the Douglas–Rachford decomposition algorithm are reported in §5.

We briefly describe our notation now. The scalar product of two vectors x and y in $\mathbb{R}^n$ is denoted by $x^T y$. All vectors are column vectors. Superscript ^{T} indicates transpose. The symbol x_+ will denote the vector with components $(x_+)_i = \max\{0, x_i\}$. The Euclidean norm of a vector x will be denoted by $\|x\|$.

2. Complex coordination. To illustrate the complex coordination paradigm, we first present the classical Dantzig–Wolfe method [1] and then describe some variants intended to exhibit greater parallel efficiency without sacrificing rate of convergence. The Schultz–Meyer method [20], [21], which also employs a complex coordination mechanism, is generalized to improve its parallel efficiency in §4.

To describe the Dantzig–Wolfe method, we denote the kth block of constraints by

$$\mathcal{B}_{[k]} := \left\{ x_{[k]} \in \mathbb{R}^{n_k} : A_{[k]} x_{[k]} = b_{[k]}, 0 \le x_{[k]} \le u_{[k]} \right\}$$

for each $k = 1, \ldots, K$. From the Minkowski Representation Theorem, each feasible point in $\mathcal{B}_{[k]}$ can be written as the sum of a convex combination of the r_k extreme points of $\mathcal{B}_{[k]}$ and a nonnegative combination of the s_k extreme rays of $\mathcal{B}_{[k]}$. Therefore, the original problem (1) is equivalent to

$$\underset{\alpha,\beta}{\text{minimize}} \quad \sum_{k=1}^{K}\left[\sum_{j=1}^{r_k}({c_{[k]}}^T x_{[k]}^j)\alpha_{k,j} + \sum_{j=1}^{s_k}({c_{[k]}}^T d_{[k]}^j)\beta_{k,j}\right]$$

$$\text{subject to} \quad \sum_{k=1}^{K}\left[\sum_{j=1}^{r_k}(D_{[k]}x_{[k]}^j)\alpha_{k,j} + \sum_{j=1}^{s_k}(D_{[k]}d_{[k]}^j)\beta_{k,j}\right] \le d, \tag{2}$$

$$\sum_{j=1}^{r_k}\alpha_{k,j} = 1 \quad \forall\, k = 1,\ldots,K$$

$$\alpha_{k,j} \ge 0, \qquad \beta_{k,j} \ge 0,$$

where for each $k = 1,\ldots,K$

$$x_{[k]}^1, x_{[k]}^2, \ldots, x_{[k]}^{r_k}$$

and

$$d_{[k]}^1, d_{[k]}^2, \ldots, d_{[k]}^{s_k}$$

are, respectively, the extreme points and the extreme rays of $\mathcal{B}_{[k]}$.

The classical Dantzig–Wolfe decomposition method solves the above minimization problem using a column generation technique. Columns with negative reduced costs (if they exist) relative to the current duals p and q_k can be obtained by solving the K independent subproblems:

$$\begin{aligned} &\underset{x_{[k]}}{\text{minimize}} && \left(c_{[k]} - {D_{[k]}}^T p\right)^T x_{[k]} - q_k \\ &\text{subject to} && x_{[k]} \in \mathcal{B}_{[k]}. \end{aligned} \tag{3}$$

In our parallel implementation on the Connection Machine CM–5, a single processor (processor 0, the master processor) is responsible for determining the new dual solution, and the remaining processors (slave processors) generate new columns with negative reduced costs. At the beginning of each major iteration, processor 0 broadcasts the new dual variable p associated with the mutual constraints to all remaining processors. For every $k = 1,\ldots,K$, the dual variables associated with the kth convex combination constraint are sent to the processor dealing with the kth block. Then, the slave processors will solve a new linear programming problem to determine if a new column with negative reduced cost exists. These columns will be sent to the master processor. Only the current basis and the new columns generated by the subproblems are kept. This drastically reduces the size of the linear programming problem that processor 0 must solve. Since the slave nodes are inactive while waiting for new dual information from the master processor, it is extremely important to control the size of the master problem and to rapidly determine a new optimal basis for the restricted master. In our implementation, MINOS 5.4 [17], [18] is used to solve both the restricted master and the subproblems. Since the feasible region for the subproblems does not change, warm start techniques can be used. Moreover, the current basis for the restricted master is feasible for the new problem.

Termination of the algorithm can be detected using a simple "reduce" operation (in Connection Machine CM–5 terminology) across the slave processors to check that

if new columns with negative reduced cost exist (i.e., if there are subproblems whose objective function is unbounded below or with strictly negative optimal value). To ensure convergence of the algorithm, special precautions must be taken to avoid cycling in the solution of the master problem. In particular, columns must be retained in the master problem if the new optimal value is identical to the optimal value achieved at the previous iteration.

It is worthwhile to note that when termination occurs, the optimal value of the master problem is also the optimal value of the original problem (1). A dual optimal solution for (1) is given by the dual optimal solution for the subproblems and the dual optimal solution in the master problem corresponding to the mutual constraints. In principle, a primal optimal solution can be also easily obtained from the optimal α, β. However, this requires saving the $x^j_{[k]}$ and $d^j_{[k]}$ vectors as well as sharing a large amount of data between the master and the slave processors. The following lemma shows an efficient parallel technique (requiring the solution of K independent subproblems) for determining a primal optimal solution.

LEMMA 2.1. *Suppose the master problem has an optimal solution $\hat{\alpha}, \hat{\beta}$ and let*

$$\hat{d}_{[k]} = \sum_{j=1}^{r_k} \left(D_{[k]} {x_{[k]}}^j \right) \hat{\alpha}_{k,j} + \sum_{j=1}^{s_k} \left(D_{[k]} {y_{[k]}}^j \right) \hat{\beta}_{k,j} \tag{4}$$

for each $k = 1, \ldots, K$. Then, a primal optimal solution of the original problem (1) *can be obtained by solving the K subproblems*

$$\begin{aligned} \underset{x_{[k]}}{\text{minimize}} \quad & {c_{[k]}}^T x_{[k]} \\ \text{subject to} \quad & A_{[k]} x_{[k]} = b_{[k]} \\ & D_{[k]} x_{[k]} \leq \hat{d}_{[k]} \\ & 0 \leq x_{[k]} \leq u_{[k]}. \end{aligned} \tag{5}$$

3. Simple coordination. Douglas–Rachford and Peaceman–Rachford decomposition methods. In contrast with the Dantzig–Wolfe method and its variants, the coordination mechanism that we consider in this section requires only summing and averaging of subproblem results.

Splitting methods (also known as alternating direction methods) were first suggested by Douglas and Rachford [2] and Peaceman and Rachford [19] for the solution of partial differential equations. A good review of these methods can be found in [16]. Later these methods were extended to finding zeros of monotone operators [15], [11] and finding Lagrangian saddlepoints in problems in solid and fluid mechanics [10], [9]. Recently, Eckstein [3] discussed various decomposition techniques for large scale optimization problems based on the alternating direction method and the connection of these methods to proximal point algorithms and partial inverse of monotone operators [22], [23] and to other "block" decomposition methods [12], [13]. In [5], Eckstein and Bertsekas present a very general convergence theory for this class of methods that does not require exact solution of the subproblems.

In the following we give a brief description of the Douglas–Rachford method and point out the differences with the Peaceman–Rachford method. More detailed discussion of these methods can be found in [3]. Given the closed proper convex functions $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ and $g : \mathbb{R}^m \to \mathbb{R} \cup \{+\infty\}$ and an $m \times n$ real matrix M,

the Douglas–Rachford splitting method for solving the convex minimization problem

$$\begin{array}{ll} \underset{x,w}{\text{minimize}} & f(x)+g(w) \\ \text{subject to} & Mx=w \end{array} \tag{6}$$

consists of constructing sequences $\{x^t\}$, $\{w^t\}$, and $\{p^t\}$, starting from arbitrary initial estimates (w^0,p^0), as follows:

$$x^{t+1}=\underset{x}{\operatorname{argmin}}\ \{f(x)+<p^t,Mx>+\tfrac{\lambda}{2}\|Mx-w^t\|^2\}, \tag{7a}$$

$$w^{t+1}=\underset{w}{\operatorname{argmin}}\ \{g(w)-<p^t,w>+\tfrac{\lambda}{2}\|Mx^{t+1}-w\|^2\}, \tag{7b}$$

$$p^{t+1}=p^t+\lambda(Mx^{t+1}-w^{t+1}), \tag{7c}$$

where λ is a positive scalar.

The following theorem summarizes the convergence of the above method.

THEOREM 3.1. *Suppose that the convex optimization problem* (6) *possesses a Karush–Kuhn–Tucker point and M has full column rank. Then, for any starting point (w^0,p^0), the iterative scheme* (7) *is well defined and the sequences $\{x^t\}$ and $\{p^t\}$ converge to primal and dual solutions of* (6). *Moreover, the sequence $\{w^t\}$ converges to $M\bar{x}$ where $\bar{x}$ is the limit of $\{x^t\}$.*

To apply the above scheme to block-angular problems, let us first define, for each $k=1,\dots,K$

$$h_{[k]}(x_{[k]}):=\begin{cases} f_{[k]}(x_{[k]}) & \text{if } A_{[k]}x_{[k]}=b_{[k]},\ 0\le x_{[k]}\le u_{[k]}, \\ +\infty & \text{otherwise} \end{cases}$$

and

$$f\left(x_{[1]},\dots,x_{[K]},d_{[1]},\dots,d_{[K]}\right):=\begin{cases} \displaystyle\sum_{i=1}^{K} h_{[k]}(x_{[k]}) & \text{if } D_{[1]}x_{[1]}\le d_{[1]},\dots,D_{[K]}x_{[K]}\le d_{[K]}, \\ +\infty & \text{otherwise,} \end{cases}$$

$$g\left(\tilde{x}_{[1]},\dots,\tilde{x}_{[K]},\tilde{d}_{[1]},\dots,\tilde{d}_{[K]}\right):=\begin{cases} 0 & \text{if } \displaystyle\sum_{i=1}^{K}\tilde{d}_{[k]}=d, \\ +\infty & \text{otherwise,} \end{cases}$$

Thus, the original block-angular problem (1) is equivalent to

$$\begin{array}{ll} \text{minimize} & f\left(x_{[1]},\dots,x_{[K]},d_{[1]},\dots,d_{[K]}\right)+g\left(\tilde{x}_{[1]},\dots,\tilde{x}_{[K]},\tilde{d}_{[1]},\dots,\tilde{d}_{[K]}\right), \\ & x_{[k]}=\tilde{x}_{[k]},\quad d_{[k]}=\tilde{d}_{[k]},\quad k=1,\dots,K. \end{array}$$

After opportune simplification, it is possible to show that the Douglas–Rachford splitting method, as outlined above, requires solving the K minimization problems:

$$\begin{array}{ll} \underset{x_{[k]},d_{[k]}}{\text{minimize}} & f_{[k]}(x_{[k]})+p_{[k]}^{t\,T}d_{[k]}+\frac{\lambda}{2}\left\|x_{[k]}-x_{[k]}^t\right\|^2+\frac{\lambda}{2}\left\|d_{[k]}-d_{[k]}^t\right\|^2 \\ \text{subject to} & A_{[k]}x_{[k]}=b_{[k]}, \\ & 0\le x_{[k]}\le u_{[k]}, \\ & D_{[k]}x_{[k]}\le d_{[k]}. \end{array} \tag{8}$$

Let $x_{[k]}^{t+1}$, $d_{[k]}^{t+1/2}$ be the optimal solution of (8). Then the new splitting of the common resource is given by

$$d_{[k]}^{t+1} = d_{[k]}^{t+\frac{1}{2}} + \frac{1}{\lambda} p_{[k]}^{t} + \frac{1}{K}\left(d - \sum_{k=1}^{K} d_{[k]}^{t+\frac{1}{2}} - \frac{1}{\lambda}\sum_{k=1}^{K} p_{[k]}^{t}\right) \tag{9}$$

and, finally, the new multipliers are

$$p_{[k]}^{t+1} = p_{[k]}^{t} + \lambda\left(d_{[k]}^{t+\frac{1}{2}} - d_{[k]}^{t+1}\right). \tag{10}$$

It should be noted that if the dual multipliers $p_{[k]}^{t}$ are the same for all the subproblems, the above formula (10) guarantees that the newly computed $p_{[k]}^{t+1}$ are again equal for all subproblems. In this case the formula for $d_{[k]}^{t+1}$ simplifies to

$$d_{[k]}^{t+1} = d_{[k]}^{t+\frac{1}{2}} + \frac{1}{K}\left(d - \sum_{k=1}^{K} d_{[k]}^{t+\frac{1}{2}}\right). \tag{11}$$

In our implementation we initially set $p_{[k]}^{0} = p^{0}$ for all $k = 1, \ldots, K$ and therefore all subproblems use the same duals. The algorithm was implemented on the Connection Machine CM–5 using the CMMD software library for interprocessor communication [25]. Each processor solves a subproblem of the form (8). After independently solving the subproblems, the new splitting $d_{[k]}^{t+1}$ must be calculated. This is achieved via a CMMD `scalar_reduce` operation that will compute $\sum_{k=1}^{K} d_{[k]}^{t+1/2}$. The result of this operation is available to all processors, which then compute the new $d_{[k]}^{t+1}$ and $p_{[k]}^{t+1}$. The coordination process among the processors thus simply reduces to computing $\sum_{k=1}^{K} d_{[k]}^{t+1/2}$ and no other information is shared except for checking termination.

The Peaceman–Rachford variant differs from the Douglas–Rachford variant only through the addition of an intermediate update of the dual multipliers and in the proximal point term used in the objective function of the subproblems. The implementation of the Peaceman–Rachford splitting method is very similar to the implementation of the Douglas–Rachford algorithm. Both methods possess linear rates of convergence [11]. The Peaceman–Rachford variant is known to be less robust and to converge under more restrictive assumptions than the Douglas–Rachford variant [10]. However, if it does converge, it converges faster. In our implementation of the Peaceman–Rachford splitting method, we found that the algorithm is very sensitive to numerical instability and we were not able to obtain results better than those obtained with the Douglas–Rachford method. Therefore, only computational results for the Douglas–Rachford method are presented in §5.

3.1. The Douglas–Rachford method for multicommodity network flow problems. In this section we discuss the application of the Douglas–Rachford splitting method to multicommodity network flow problems. While general methods such as those used in MINOS may be employed to solve the subproblem (8), we are also interested in taking advantage of special structure to obtain more efficient methods for the important class of problems in which the matrices $A_{[k]}$ in (1) are node-arc incidence matrices and $D_{[k]}$ are matrices whose columns are either columns of the identity or zero.

The subproblem (8) is equivalent to

(12)

$$\begin{aligned} \underset{x_{[k]}}{\text{minimize}} \quad & f_{[k]}(x_{[k]}) + \tfrac{\lambda}{2}\left\|x_{[k]} - x^t_{[k]}\right\|^2 + \left[\min_{D_{[k]}x_{[k]} \le d_{[k]}} \left({p^t_{[k]}}^T d_{[k]} + \tfrac{\lambda}{2}\left\|d_{[k]} - d^t_{[k]}\right\|^2\right)\right] \\ \text{subject to} \quad & A_{[k]}x_{[k]} = b_{[k]}, \\ & 0 \le x_{[k]} \le u_{[k]}. \end{aligned}$$

The minimization over $d_{[k]}$ for the problem above can be carried out analytically yielding an optimal solution point

$$d^{t+\frac{1}{2}}_{[k]} = d^t_{[k]} - \frac{1}{\lambda}p^t_{[k]} + \left(D_{[k]}x_{[k]} - \left(d^t_{[k]} - \tfrac{1}{\lambda}p^t_{[k]}\right)\right)_+. \tag{13}$$

Substituting the above in (12) and disregarding the additive constant we get

$$\begin{aligned} \underset{x_{[k]}}{\text{minimize}} \quad & f_{[k]}(x_{[k]}) + \tfrac{\lambda}{2}\left\|x_{[k]} - x^t_{[k]}\right\|^2 + \tfrac{\lambda}{2}\left\|\left(D_{[k]}x_{[k]} - d^t_{[k]} + \tfrac{1}{\lambda}p^t_{[k]}\right)_+\right\|^2_2 \\ \text{subject to} \quad & A_{[k]}x_{[k]} = b_{[k]}, \\ & 0 \le x_{[k]} \le u_{[k]}, \end{aligned} \tag{14}$$

which, due to the special nature of the matrix $D_{[k]}$, is a separable function as long as the function $f_{[k]}$ is separable. Therefore, the subproblems are nonlinear separable network flow problems on the same graph as the original multicommodity problem. We can approximate the separable functions by a sequence of piecewise-linear overestimates as described in [14]. The resulting approximation to (14) can be solved using standard techniques for linear network flow problems after associating an arc having appropriate cost and capacity with each segment in the piecewise-linear approximation. The optimal value of any such approximation provides an upper bound on the true optimal value of (14). Lower bounds can be obtained using Lagrangian relaxation techniques.

4. Multicoordinator methods. This section describes a generalization of the Schultz–Meyer method to multiple coordinating problems that may be solved in parallel. As in Schultz–Meyer, there will be K subproblems where K is the number of blocks of the problem. However, instead of a single coordination problem, we consider here an approach in which there will be P coordinators (each associated with K/P or more subproblems). A related multicoordinator approach is considered by Mangasarian and Ferris [6]–[8].

The P coordinating problems are solved in parallel, each producing a proposed next iterate. The point produced by the coordinator that has the best (least) objective value is then chosen as the next iterate. In §4.2 we consider the simplest case with $P = K$ and one coordinator for each subproblem, since this is the easiest approach for developing the convergence theory. This scheme is referred to as "single-variable" multicoordination. In §4.3, two other types of multicoordination are presented. The first approach, called "group coordination" is similar to "single-variable" multicoordination except each processor is responsible for a group of subproblems and solves a coordination problem with as many variables as the number of subproblems in its

group. The second approach, "block-plus-group" multicoordination, uses coordinators that each deal explicitly with a certain block of variables plus a set of stepsize variables corresponding to the subproblems in its group.

We start by introducing the problem we wish to solve:

$$\min_x \quad f(x) := c(x) + \tau\rho(\theta - Dx) \quad \text{subject to} \quad x \in \mathcal{B}, \tag{15}$$

where $c(x)$ represents the linear or nonlinear objective function of the block-angular problem, $\tau > 0$ is the penalty parameter, and ρ is a barrier function on the "shifted" coupling constraint $Dx \leq \theta$ (for details on the determination of θ, see [21]); finally

$$\mathcal{B} := \left\{ x \in \mathbb{R}^N, \;\; x^T = \left[x_{[1]}^T, x_{[2]}^T, \ldots, x_{[K]}^T\right], \right.$$
$$\left. x_{[k]} \in \mathbb{R}^{n_k}, \;\; A_{[k]}x_{[k]} = b_{[k]}, \;\; 0 \leq x_{[k]} \leq u_{[k]} \;\; \forall k = 1, \ldots, K \right\}.$$

We assume that f is C^1 and convex on $\mathcal{B}$.

4.1. Convex quadratic subproblems. Define a *decoupled resource allocation* for $\mathcal{B}$ to be an interval valued map R (essentially a trust region box "centered" around the argument x) with

$$R(x,t) := \{z \in \mathbf{R}^N \mid \underline{\mathcal{R}}(x,t) \leq z \leq \overline{\mathcal{R}}(x,t)\},$$

where

$$\forall x \in \mathcal{B} \quad 0 \leq \underline{\mathcal{R}}(x,t) \leq x \leq \overline{\mathcal{R}}(x,t) \leq u, \tag{16}$$

for any bounded sequence $\{x^t\} \subseteq \mathcal{B}$

$$\bigcup_{t=0}^{\infty} \mathcal{R}(x^t,t) \quad \text{is bounded} \tag{17}$$

and for any $z \in \mathcal{B}$ and any bounded sequence $\{x^t\} \subseteq \mathcal{B}$ with

$$\alpha^t := \max\{\alpha \mid 0 \leq \alpha \leq 1 \text{ and } x^t + \alpha(z - x^t) \in \mathcal{R}(x^t,t)\}, \tag{18}$$

we have $\liminf_{t\to\infty} \alpha^t > 0$.

The subproblem for determining the search direction generated by the kth block then is of the form:

$$\begin{array}{ll} \underset{y^t_{[k]}}{\text{minimize}} & g^t_{[k]}\left(y^t_{[k]} - x^t_{[k]}\right) + \left(y^t_{[k]} - x^t_{[k]}\right)^T G^t_{[k]}\left(y^t_{[k]} - x^t_{[k]}\right) \\ \text{subject to} & A_{[k]}y^t_{[k]} = b_{[k]} \\ & \underline{\mathcal{R}}(x^t,t)_{[k]} \leq y^t_{[k]} \leq \overline{\mathcal{R}}(x^t,t)_{[k]}. \end{array} \tag{19}$$

Where it is assumed that if g^t is the concatenation of $g^t_{[k]}$

$$\lim_{t\to\infty} ||g^t - \nabla f(x^t)|| = 0, \tag{20}$$

and if G^t is the block diagonal matrix with $G^t_{[k]}$ for the kth block

$$\exists \beta > 0 \text{ such that } (\forall t)(\forall x) \quad 0 \leq x^T G^t x \leq \beta \|x\|^2. \tag{21}$$

In the original Schultz–Meyer method, coordination of the subproblem solution is accomplished by solving the nonlinear optimization problem

$$\begin{aligned} \underset{w\in\mathbb{R}^K}{\text{minimize}} \quad & f\left(x^t_{[1]} + \left(y^t_{[1]} - x^t_{[1]}\right) w_1, \ldots, x^t_{[K]} + \left(y^t_{[K]} - x^t_{[K]}\right) w_K\right) \\ \text{subject to} \quad & 0 \le x^t_{[k]} + \left(y^t_{[k]} - x^t_{[k]}\right) w_k \le u_{[k]}, \qquad k = 1, \ldots, K. \end{aligned}$$

Here we consider generalizations based on multiple coordination problems and prove convergence by focusing on properties related to the best subproblem at each iteration.

Let $j(t)$ be the index of the subproblem that produces the least subproblem objective and $y^t_{[j(t)]}$ be the direction produced by that subproblem. Define $\overline{y}^t$ blockwise as

$$\overline{y}^t_{[k]} = \begin{cases} y^t_{[k]} & \text{if } k = j(t), \\ x^t_{[k]} & \text{otherwise.} \end{cases} \tag{22}$$

THEOREM 4.1 (Sufficiency of search directions). *Suppose that* $\{x^t\} \subset \mathcal{B} \cap \operatorname{dom} f$, $x^t \to \tilde{x} \in \mathcal{B} \cap \operatorname{dom} f$, *and* $\overline{y}^t$ *is as above. Also suppose there exists an accumulation point* $\tilde{y}$ *of* $\{\overline{y}^t\}$ *satisfying*

$$\nabla f(\tilde{x})(\tilde{y} - \tilde{x}) \ge 0. \tag{23}$$

Then

$$\forall z \in \mathcal{B} \quad \nabla f(\tilde{x})(z - \tilde{x}) \ge 0.$$

Proof. Without loss of generality thin the sequence so that $\overline{y}^t \to \tilde{y}$. Then from (20), (21), and (23) and the definition of $\overline{y}^t_{[k]}$, it follows that for any block k

$$(\forall \epsilon > 0) \quad \exists t_\epsilon \text{ such that } (\forall t > t_\epsilon)$$

$$g^t_{[k]}\left(y^t_{[k]} - x^t_{[k]}\right) + \left(y^t_{[k]} - x^t_{[k]}\right)^T G^t_{[k]}\left(y^t_{[k]} - x^t_{[k]}\right) > -\epsilon. \tag{24}$$

Now suppose for some $z \in \mathcal{B}$

$$\nabla f(\tilde{x})(z - \tilde{x}) = -\zeta < 0.$$

Still further thin the sequence so that $G^t \to \tilde{G}$. Define α^t as in (18) and $\tilde{\alpha} := \liminf_t \alpha^t > 0$. Then $\lambda \in (0,1)$ independent of t may be picked such that

$$\lambda\tilde{\alpha}(z - \tilde{x})^T \tilde{G}(z - \tilde{x}) < \zeta/2,$$

then

$$\lambda\tilde{\alpha}\nabla f(\tilde{x})(z - \tilde{x}) + \lambda^2\tilde{\alpha}^2(z - \tilde{x})^T\tilde{G}(z - \tilde{x}) < -\frac{\lambda\zeta\tilde{\alpha}}{2} < 0.$$

Hence $(\exists \tilde{t} > 0)$ such that $(\forall t > \tilde{t})$

$$g^t[\lambda\tilde{\alpha}(z - x^t)] + [\lambda\alpha^t(z - x^t)]^T G^t[\lambda\alpha^t(z - x^t)] < -\frac{\lambda\zeta\tilde{\alpha}}{4} < 0.$$

This implies there exists a block m such that

$$g^t_{[m]}\left[\lambda\tilde{\alpha}(z_{[m]} - x^t_{[m]})\right] + \left[\lambda\alpha^t(z_{[m]} - x^t_{[m]})\right]^T G^t_{[m]}\left[\lambda\alpha^t(z_{[m]} - x^t_{[m]})\right] < -\frac{\lambda\zeta\tilde{\alpha}}{4K} < 0,$$

where K is the number of blocks. The above contradicts (24) and hence the conclusion of the theorem is valid. □

4.2. Single-variable coordination problems. In this section each coordination problem involves only a single variable and there is a one-to-one correspondence between the coordinators and the subproblems. The subproblems generate descent directions for their block (or demonstrate that none exist) and then each subproblem communicates the descent direction to the coordinator responsible for that particular subproblem. The following describes the coordinator and the properties of the solution of the coordinator.

As before, let $y^t_{[k]}$ be the direction produced by the kth subproblem. The kth coordinator problem is

$$\begin{array}{ll} \underset{w_k \in \mathbb{R}}{\text{minimize}} & f\left(x^t_{[1]}, \ldots, x^t_{[k-1]}, x^t_{[k]} + \left(y^t_{[k]} - x^t_{[k]}\right) w_k, x^t_{[k+1]}, \ldots, x^t_{[K]}\right) \\ \text{subject to} & 0 \le x^t_{[k]} + \left(y^t_{[k]} - x^t_{[k]}\right) w_k \le u_{[k]}. \end{array} \tag{25}$$

It should be noted that the above constraints are equivalent to bound constraints on w_k. Let

$$E(k,t) := \left\{ x \in \mathbb{R}^N : x_{[l]} = x^t_{[l]} \text{ for } l \neq k, \;\; x_{[k]} = (1-\lambda)x^t_{[k]} + \lambda y^t_{[k]}, \;\; 0 \le \lambda \le 1 \right\}.$$

Observe that $E(k,t)$ is a subset of the feasible region for the coordinator. Coordinator k is required to compute an approximate solution of (25) $x^{k,t+1}$ satisfying

$$f(x^{k,t+1}) - f(x^t) \le \mu \left[\min_{x \in E(k,t)} f(x) - f(x^t) \right], \tag{26}$$

where $\mu \in (0,1)$ is independent of t. To obtain a computationally checkable condition for (26), suppose that the zero vector is not optimal for coordinator k, and let

$$f^t_k(\xi) = f\left(x^t_{[1]}, \ldots, x^t_{[k-1]}, x^t_{[k]} + \left(y^t_{[k]} - x^t_{[k]}\right)\xi, x^t_{[k+1]}, \ldots, x^t_{[K]}\right).$$

Then

$$\mu \left[\min_{x \in E(k,t)} f(x) - f(x^t) \right] = \mu \left[\min_{x \in E(k,t)} f(x) - f^t_k(\mathbf{0}) \right]$$

and

$$\min_{x \in E(k,t)} f(x) = \min_{0 \le \xi \le 1} f^t_k(\xi).$$

Since f is convex, $f^t_k(\xi)$ is also convex and therefore for every $\bar{\xi} \in \mathbb{R}$

$$f^t_k(\xi) \ge f^t_k(\bar{\xi}) + (\xi - \bar{\xi}) \nabla f^t_k(\bar{\xi})$$

and

$$\min_{0 \le \xi \le 1} f^t_k(\xi) \ge f^t_k(\bar{\xi}) + \min\left\{ \nabla f^t_k(\bar{\xi})(0 - \bar{\xi}), \nabla f^t_k(\bar{\xi})(1 - \bar{\xi}) \right\}$$

since the linear functional takes its minimum over a line segment at one end. From the above inequality, we obtain

$$\begin{aligned} \mu \left[\min_{x \in E(k,t)} f(x) - f(x^t) \right] &= \mu \left[\min_{x \in E(k,t)} f(x) - f^t_k(\mathbf{0}) \right] \\ &\ge \mu \big[f^t_k(\bar{\xi}) - f^t_k(0) + \\ &\qquad \min\left\{ \nabla f^t_k(\bar{\xi})(0 - \bar{\xi}), \nabla f^t_k(\bar{\xi})(1 - \bar{\xi}) \right\} \big]. \end{aligned}$$

Thus, if $\{\xi^i\}$ is a sequence of reals converging to a minimizer of f_k^t on $[0,1]$, then

$$\mu\Big[f_k^t(\xi^i) - f_k^t(0) + \min\left\{\nabla f_k^t(\xi^i)(0-\xi^i), \nabla f_k^t(\xi^i)(1-\xi^i)\right\}\Big]$$

converges to

$$\mu\left[\min_{x\in E(k,t)} f(x) - f_k^t(0)\right].$$

Therefore, the sufficient decrease condition is satisfied by any point $x^{k,t+1}$ corresponding to $w_k = \xi^i$ (see (25)) such that

$$f(x^{k,t+1}) - f(x^t) \le \mu\Big[f_k^t(\xi^i) - f_k^t(0) + \min\left\{\nabla f_k^t(\xi^i)(0-\xi^i), \nabla f_k^t(\xi^i)(1-\xi^i)\right\}\Big].$$

At each iteration, K subproblems are solved, each producing a search direction for the corresponding block. The direction determined by the kth subproblem is communicated to the kth coordinator (recall that here $P = K$ and that there is one coordinator per subproblem) and the kth coordinator produces $x^{k,t+1}$. Then x^{t+1} (the new x iterate) is chosen as the point producing the best value amongst the $x^{k,t+1}$ and therefore

$$f(x^{t+1}) - f(x^t) \le \mu\left[\min_{x\in E(j,t)} f(x) - f(x^t)\right], \tag{27}$$

where μ is the same as in (26).

THEOREM 4.2 (Convergence of decomposition method using single-variable multicoordinators). *Suppose that*

1. *f is convex and C^1 on $\mathcal{B}$,*
2. *$x^0 \in \mathcal{B} \cap \operatorname{dom} f$,*
3. *$y_{[k]}^t$ is the solution to the kth subproblem where* (16) *and* (17) *hold for some decoupled resource allocation R and*
4. *$x^{t+1} \in \mathcal{B}$ satisfies the coordinator condition* (27) *for all t.*

Then either $\lim_{t\to\infty} f(x^t) = -\infty$ or $\lim_{t\to\infty} f(x^t) = \tilde{f} > -\infty$, and in the latter case the limit points of x^t are optimal solutions of the original problem (1).

Proof. From (27) we observe that the sequence $f(x^t)$ is decreasing. Therefore either the function values diverge to $-\infty$ or else they have a finite limiting value $\tilde{f}$. Suppose the latter is the case. Let $\tilde{x}$ be any limit point of $\{x^t\}$. Since f has the property of diverging to $+\infty$ at the barrier and the function values are decreasing, it follows that $\tilde{x} \in \operatorname{dom} f$. We see that $\lim_{t\to\infty} f(x^{t+1}) - f(x^t) = 0$. Consider a subsequence $\{x^{\sigma(t)}\} \subseteq \{x^t\}$ with $x^{\sigma(t)} \to \tilde{x}$.

Let $\overline{y}^{\sigma(t)}$ be defined as in (22) and denote $E(j(t),t)$ by $E(t)$. Now by the boundedness of the resource allocation (17), we have that $\{\overline{y}^{\sigma(t)}\}$ is bounded. Hence the sequence may further be thinned to obtain $\overline{y}^{\sigma(t)} \to \tilde{y}$. Now for all $\lambda \in (0,1]$ the following hold:

$$\begin{aligned} f\left((1-\lambda)x^{\sigma(t)} + \lambda\overline{y}^{\sigma(t)}\right) &\ge \left[\min_{x\in E(\sigma(t))} f(x) - f(x^{\sigma(t)})\right] \\ &\ge \frac{1}{\mu}\Big[f(x^{\sigma(t)+1}) - f(x^{\sigma(t)})\Big]. \end{aligned}$$

The first inequality follows from the definition of $E(t)$ and the second from (27). Since the right-hand side converges to 0 as $t \to \infty$, we have $f((1-\lambda)\tilde{x} + \lambda\tilde{y}) - f(\tilde{x}) \ge 0$. Taking the limit as λ approaches 0 we get $\nabla f(\tilde{x})(\tilde{y} - \tilde{x}) \ge 0$. The conclusion follows from Theorem 4.1. □

4.3. Group and block-plus-group multicoordination. In this section two multicoordinator coordination schemes are presented as alternatives to single-variable coordinators: "group" coordinators and "block-plus-group" coordinators. There are no changes in the subproblems.

Group coordinators use results from a group of subproblems instead of just a single subproblem. Let $\Gamma_p \subseteq \{1,\dots,K\}$ denote the group of subproblems for which coordinator p is responsible. We assume that $\bigcup_{p=1}^{P} \Gamma_p = \{1,\dots,K\}$. However, the subsets need not be disjoint. An efficient mechanism for forming groups in a parallel environment would be to assemble them and begin the solution of the corresponding coordination problem as subproblem solutions become available. Synchronization delay may be reduced by providing for smaller group sizes for the results of subproblems that terminate late.

The pth coordinator problem is

$$\begin{aligned} \underset{w\in\mathbb{R}^K}{\text{minimize}} \quad & f\left(x^t_{[1]} + \left(y^t_{[1]} - x^t_{[1]}\right) w_1, \dots, x^t_{[K]} + \left(y^t_{[K]} - x^t_{[K]}\right) w_K\right) \\ \text{subject to} \quad & 0 \le x^t_{[k]} + \left(y^t_{[k]} - x^t_{[k]}\right) w_k \le u_{[k]}, \quad k \in \Gamma_p, \\ & w_j = 0, \qquad j \notin \Gamma_p. \end{aligned}$$

Although, formally stated as a minimization problem in the K-dimensional space, the coordination problem is an optimization problem in the much smaller dimensional space $\mathbb{R}^{|\Gamma_p|}$. (If $|\Gamma_p| = K$, then the preceding problem is precisely the original Schultz–Meyer coordination problem.)

Let $i_p \in \Gamma_p$ be the index of the subproblem that produced the best descent direction among those in Γ_p. We require that the new point computed by the pth coordinator satisfy the inequality

$$f(x^{p,t+1}) - f(x^t) \le \mu \left[\min_{x\in E(i_p,t)} f(x) - f(x^t) \right]. \tag{28}$$

(Note that the above is certainly satisfied at optimality since $E(i_p,t)$ is a subset of the feasible region for the pth coordinator.) As in the single coordination case, a computationally checkable condition that guarantees the above inequality can be easily determined. Again, we choose x^{t+1} so that

$$f(x^{t+1}) \le f(x^{p,t+1}) \quad \forall p = 1,\dots,P,$$

and hence

$$f(x^{t+1}) - f(x^t) \le \mu \left[\min_{x\in E(j(t),t)} f(x) - f(x^t) \right],$$

where $j(t)$ is again the index of the subproblem that produces the best direction among all subproblems (19). This is the crucial inequality needed in the convergence proof of the decomposition method. (Note, in fact, that after the coordination problem $j(t)$ is solved, later coordination problems may be truncated or eliminated once the dominance of subproblem $j(t)$ is verified.) Therefore, Theorem 4.2 continues to hold with this new choice of the coordination mechanism.

A "block-plus-group" coordinator handles a particular original block of variables explicitly as well as stepsize variables for directions provided by subsets of the subproblems. More formally, for $k = 1,\dots,K$, let Γ_k be a subset of $\{1,\dots,K\} - \{k\}$.

As before, we assume that $\bigcup_{i=1}^{K} \Gamma_k = \{1, \ldots, K\}$, but we do not require the subsets to be disjoint.

The kth coordinator problem is

$$\begin{aligned} \underset{w, x_{[k]}}{\text{minimize}} \quad & f\left(x^t_{[1]} + \left(y^t_{[1]} - x^t_{[1]}\right) w_1, \ldots, x^t_{[k-1]} + \left(y^t_{[k-1]} - x^t_{[k-1]}\right) w_{k-1}, \right. \\ & \left. x_{[k]}, x^t_{[k+1]} + \left(y^t_{[k+1]} - x^t_{[k+1]}\right) w_{k+1}, \ldots, x^t_{[K]} + \left(y^t_{[K]} - x^t_{[K]}\right) w_K \right) \\ \text{subject to} \quad & A_{[k]} x_{[k]} = b_{[k]} \\ & 0 \le x_{[k]} \le u_{[k]} \\ & 0 \le x^t_{[l]} + \left(y^t_{[l]} - x^t_{[l]}\right) w_l \le u_{[l]}, \quad l \in \Gamma_k, \\ & w_l = 0, \qquad l \notin \Gamma_k. \end{aligned}$$

With this type of coordinator, we may alternate improvements to $x_{[k]}$ and w, taking advantage of special structure of the block constraints $\mathcal{B}_{[k]}$ during iterations in which w is fixed. Let i_k be the index of the subproblem producing the best direction among those in $\Gamma_k \cup \{k\}$. We again require that the new point produced by the kth coordinator satisfy the inequality

$$f(x^{k,t+1} - f(x^{k,t}) \le \mu \left[\min_{x \in E(i_k,t)} f(x) - f(x^{k,t}) \right]. \tag{29}$$

A computationally checkable condition for (29) can be constructed as done previously. Again, we choose x^{t+1} as the best (in terms of objective function value) of the proposals $x^{k,t+1}$ from the various coordinators and hence the inequality

$$f(x^{t+1}) - f(x^t) \le \mu \left[\min_{x \in E(j(t),t)} f(x) - f(x^t) \right]$$

is still satisfied and Theorem 4.2 holds for the block-plus-group coordination scheme.

5. Computational results. In this section we present computational results for the various coordination mechanisms discussed in previous sections (except for the multicoordinator methods, which are currently being implemented). All results were obtained on the Connection Machine CM–5 running the CMOST Version 7.2 S3 operating system. The CMMD software library (version 3.02) was used for interprocessor communication.

The subproblems arising in the simple-coordination Douglas–Rachford splitting method, as well as the subproblems and the master problem for the Dantzig–Wolfe decomposition method, were solved using MINOS 5.4.

We tested the Douglas–Rachford splitting method and the Dantzig–Wolfe decomposition algorithm on multicommodity network optimization problems randomly generated with MNETGEN.

Table 1 shows the problem dimension, the number of commodities (blocks), the number of nodes and arcs for each subproblem, the size of the equivalent linear program, and the number of coupling constraints. The problem name is derived from the number of nodes and the number of commodities. The number of processors used in our parallel algorithms is equal to the number of blocks for the Douglas–Rachford splitting method and the number of blocks +1 for the Dantzig–Wolfe algorithm. It should be noted that, although we tested the algorithms on multicommodity network optimization problems, the code we wrote can handle general block-angular linear and separable convex quadratic problems.

TABLE 1
Test problem characteristics.

Problem	Blocks	Subproblem size		Coupling	Equivalent LP size	
		Nodes	Arcs	constraints	Constraints	Variables
100.4	4	100	218	129	529	872
100.8	8	100	229	131	931	1832
100.16	16	100	233	157	1757	3728
100.32	32	100	224	130	3330	7168
200.4	4	200	423	252	1052	1692
200.8	8	200	449	277	1877	3592
200.16	16	200	453	287	3487	7248
300.4	4	300	678	409	1609	2712
300.8	8	300	671	408	2808	5368
300.16	16	300	661	410	5210	10576

TABLE 2
Results for the modified Dantzig–Wolfe decomposition method.

Problem	Iterations		Time
	Phase I	Phase II	secs.
100.4	4	10	3.4
100.8	3	11	4.7
100.16	3	10	7.3
200.4	5	13	11.2
200.8	4	34	39.4
200.16	5	34	67.4
300.4	11	240	378.5
300.8	7	115	298.3
300.16	8	75	340.5

Table 2 reports the number of iterations for both Phase 1 and Phase 2 of the Dantzig–Wolfe decomposition method. In our implementation, a full set of artificial variables was introduced and the initial basis consisted only of these artificial variables. As we expected, relatively few iterations are needed to obtain feasible solutions with subproblem optimality tolerance of 10^{-9}. (Similarly, as reported in [21], the Schultz–Meyer method required on the order of 50 iterations to generate highly accurate solutions for linear multicommodity problems.) However, the coordination mechanism running on a single processor does not allow us to achieve very high speedups. This suggests that it may be worthwhile to develop multicoordination approaches for the Dantzig–Wolfe method analogous to those described in §4 for the Schultz–Meyer method.

Table 3 reports the number of iterations required to achieve a solution accuracy of approximately 10^{-4} in satisfying the Karush–Kuhn–Tucker conditions for the Douglas–Rachford scheme for the block-angular linear programs. Observe that the iteration count increases rapidly with problem size in spite of the fact that a number of heuristics have been incorporated to decrease the number of iterations. In particular, we found that a good choice of the initial value of the parameter λ in (9) and a better updating scheme can drastically reduce the number of iterations needed to achieve the prescribed accuracy. In our implementation we used different values λ_1 (multiplying the term $\left\|x_{[k]} - x^t_{[k]}\right\|^2$) and λ_2 (for $\left\|d_{[k]} - d^t_{[k]}\right\|^2$) for the two proximal terms. More complicated schemes are also possible with distinct values of

TABLE 3
Computational results for Douglas–Rachford decomposition method on linear programs.

Problem	Iterations	Solution accuracy
100.4	70	3.53e–04
100.8	200	4.38e–04
100.16	145	4.63e–04
100.32	235	4.36e–04
200.4	50	1.14e–04
200.8	170	4.94e–04
200.16	320	3.49e–04
300.4	155	3.52e–04
300.8	225	4.19e–04
300.16	935	2.77e–04

λ_1 for each subproblem. Larger values of λ_1 force the new point $x_{[k]}^{t+1}$ to be closer to current point $x_{[k]}$. Similarly, larger values of λ_2 force the new splitting of the common resource d to be very similar to the current splitting, while smaller values of λ_2 allow larger variation in the splitting. However, the situation is reversed for the updating of the multiplier p^t. In fact, larger values of λ_2 allow p^t to change substantially, while a small value of λ_2 allows only small adjustments of p^t. The initial values of λ_1 and λ_2 were chosen proportional to the largest (in absolute value) coefficient in the corresponding linear objective function terms. During the algorithm, the values of λ_1 and λ_2 were modified until they reached a common final value. The initial $x_{[k]}^0$ was chosen as the solution of the linear program without the coupling constraints. The initial $d_{[k]}^0$ was chosen proportional to $D_{[k]}x_{[k]}^0$ with the additional requirement that $d_{[1]}^0 + d_{[2]}^0 + \cdots + d_{[K]}^0 = d$. Finally, the initial p^0 is chosen proportional to the magnitude of the violation of the coupling constraints for the initial $x_{[k]}^0$.

The solution times for the Douglas–Rachford algorithm for our test problem were 3–10 times greater than those of Dantzig–Wolfe in spite of the looser stopping criteria for the Douglas–Rachford scheme. However, the current timers on the Connection Machine CM–5 are somewhat inaccurate in the version of the operating system we used, and tuning of the Douglas–Rachford code via such items as advanced starting bases could substantially reduce this performance gap.

We also experimented with the Douglas–Rachford decomposition method for block-angular separable strictly convex quadratic programs. The objective function used in our tests is given by

$$f_{[1]}(x_{[1]}) + f_{[2]}(x_{[2]}) + \cdots + f_{[K]}(x_{[K]}),$$

where for all $k = 1, \ldots, K$

$$f_{[k]}(x_{[k]}) = {c_{[k]}}^T x_{[k]} + \frac{r}{2}\left\|x_{[k]}\right\|^2.$$

Table 4 reports the number of iterations needed to achieve an accuracy of at least 0.5×10^{-3} for various values of the quadratic factor $r > 0$.

The number of outer iterations is drastically reduced for quadratic programs (as compared to linear programs). Similar results were reported by Eckstein [4] for the same splitting method applied to single-commodity network optimization problems. This opens the possibility of more efficiently solving linear programming problems

TABLE 4
Comparison of number of iterations of Douglas–Rachford decomposition method on linear and quadratic programs.

Problem	$r = 0$	$r = 0.01$	$r = 0.1$	$r = 1$
100.4	70	20	10	15
100.8	200	40	45	25
100.16	145	50	50	30
100.32	235	35	50	25
200.4	50	20	15	15
200.8	170	40	65	30
200.16	320	95	70	40
300.4	155	35	80	30
300.8	225	75	65	35
300.16	935	130	70	40

by solving a sequence of strongly convex quadratic problems obtained by adding a proximal point term to the linear objective function.

Finally, we wrote a specialized code for the Douglas–Rachford decomposition scheme for multicommodity network optimization problems that uses network techniques to solve the piecewise-quadratic network optimization subproblems. The overall solution times with this approach were comparable to those obtained using MINOS. While we had expected faster times with the network approach, the large number of degenerate pivots required by the network code at each major iteration led to a similar solution time. Better methods of generating starting bases may lead to significant improvements in those times.

6. Conclusions. We have investigated a variety of coordinating mechanisms ranging from simple averaging processes to multiple optimization problems. Our computational experience demonstrates that while the highly parallelizable simple coordination of Douglas–Rachford splitting works well for block-angular problems with large quadratic terms in the objective, methods that use a single relatively difficult optimization problem for coordination (such as Dantzig–Wolfe and Schultz–Meyer) yield faster convergence for linear problems at the expense of thereby introducing a potential serial bottleneck. The new multicoordination methods proposed here are thus especially promising for linear block-angular problems in that they eliminate that serial bottleneck, help to alleviate synchronization delays associated with the termination of the subproblem phase, and provide a broad and flexible range of sizes for the optimization problems used for coordination.

REFERENCES

[1] G. DANTZIG AND P. WOLFE, *Decomposition principle for linear programs*, Oper. Res., 8 (1960), pp. 101–111.

[2] J. DOUGLAS AND H. RACHFORD JR., *On the numerical solution of heat conduction problems in two- and three-space variables*, Trans. Amer. Math. Soc., 82 (1956), pp. 421–439.

[3] J. ECKSTEIN, *Splitting Methods for Monotone Operators with Applications to Parallel Optimization*, Ph.D. thesis, Department of Civil Engineering, Massachussetts Institute of Technology, Cambridge, 1989.

[4] ———, *The alternating step method for monotropic programming on the Connection Machine* CM–2, ORSA J. Comput., 5 (1993), pp. 293–318.

[5] J. ECKSTEIN AND B. BERTSEKAS, *On the Douglas–Rachford splitting method and the proximal point method for maximal monotone operators*, Math. Programming, Ser. A, 55 (1992), pp. 293–318.

[6] M. FERRIS, *Parallel constraint distribution for convex quadratic programs*, Tech. Report 1009, Computer Sciences Department, University of Wisconsin, Madison, 1991; Math. Oper. Res., to appear.

[7] M. FERRIS AND O. MANGASARIAN, *Parallel constraint distribution*, SIAM J. Optim., 1 (1991), pp. 487–500.

[8] ———, *Parallel variable distribution*, Tech. Report 1175, Computer Sciences Department, University of Wisconsin, Madison, 1993.

[9] M. FORTIN AND R. GLOWINSKI, *Augmented Langrangian methods. Applications to the numerical solution of boundary-value problems*, Studies in Math. Appl. 15, North-Holland, Amsterdam, New York, 1983.

[10] ———, *On decomposition-coordination methods using an Augmented Lagrangian*, in Augmented Langrangian methods. Applications to the numerical solution of boundary-value problems, Studies in Math. Appl. 15, North-Holland, Amsterdam, New York, 1983.

[11] D. GABAY, *Applications of the method of multipliers to variational inequalities*, in Augmented Langrangian methods. Applications to the numerical solution of boundary-value problems, Studies in Math. Appl. 15, North-Holland, Amsterdam, New York, 1983.

[12] Y. G. GOLSHTEIN, *The block method of convex programming*, Soviet Math. Dokl., 33 (1986), pp. 584–587.

[13] ———, *A general approach to decomposition of optimization systems*, Soviet J. Computer Sys. Sci., 3 (1987), pp. 105–114.

[14] P. KAMESAM AND R. MEYER, *Multipoint methods for separable nonlinear networks*, Math. Programming Study, 22 (1984), pp. 185–205.

[15] P. LIONS AND B. MERCIER, *Splitting algorithms for the sum of two nonlinear operators*, SIAM J. Numer. Anal., 16 (1979), pp. 964–979.

[16] G. MARCHUK, *Splitting and alternating direction methods*, in Handbook of Numerical Analysis, P. Ciarlet and J. Lions, eds., North-Holland, Berlin, 1990.

[17] B. MURTAGH AND M. SAUNDERS, *MINOS* 5.1 *User's Guide*, Tech. Report SOL 83.20R, Stanford University, Stanford, CA, 1987.

[18] ———, *MINOS* 5.4 *Release Notes, Appendix to MINOS* 5.1 *user's guide*, Tech. Report, Stanford University, Stanford, CA, 1992.

[19] D. PEACEMAN AND H. RACHFORD JR., *The numerical solution of parabolic and elliptic differential equations*, SIAM J., 3 (1955), pp. 28–42.

[20] G. SCHULTZ, *Barrier Decomposition for the Parallel Optimization of Block-Angular Programs*, Ph.D. thesis, University of Wisconsin, Madison, 1991.

[21] G. SCHULTZ AND R. MEYER, *An interior point method for block angular optimization*, SIAM J. Optim., 1 (1991), pp. 583–602.

[22] J. SPRINGARN, *Partial inverse of a monotone operator*, Appl. Math. Optim., 10 (1983), pp. 247–265.

[23] ———, *Applications of the method of partial inverses to convex programming: decomposition*, Math. Programming, 32 (1985), pp. 199–223.

[24] THINKING MACHINES CORPORATION, *The Connection Machine* CM-5 *Technical Summary*, Cambridge, MA, 1991.

[25] ———, CMMD *Reference Manual, Version* 3.0, Cambridge, MA, 1993.

SIAM J. OPTIMIZATION
Vol. 4, No. 4, pp. 794–814, November 1994
© 1994 Society for Industrial and Applied Mathematics
007

PARALLEL BRANCH-AND-BOUND ALGORITHMS FOR GENERAL MIXED INTEGER PROGRAMMING ON THE CM-5*

JONATHAN ECKSTEIN†

Abstract. This "proof of concept" paper describes parallel solution of general mixed integer programs by a branch-and-bound algorithm on the CM-5 multiprocessing system. It goes beyond prior parallel branch-and-bound work by implementing a reasonably realistic general-purpose mixed integer programming algorithm, as opposed to a specialized method for a narrow class of problems. It shows how to use the capabilities of the CM-5 to produce an efficient parallel implementation employing centrally controlled search, achieving near-linear speedups using 64–128 processors on a variety of difficult problems derived from real applications. In concrete terms, a problem requiring half an hour to solve on a SPARC–2 workstation might be solved in 15–20 seconds, and a problem originally taking a week might be reduced to about an hour. Central search control does have limitations, and some final computational experiments begin to address the merits of more decentralized options.

Key words. mixed integer programming, branch and bound, parallel computing

AMS subject classifications. 90C11, 68U99

1. Introduction.

This paper describes the implementation of a branch-and-bound algorithm for general mixed integer programming on the CM-5 multiprocessor system [42]. Because integer programming problems are difficult and commonplace, and implicit enumeration procedures easily decompose into highly independent subtasks, integer programming has been a popular suggested application of parallel computing since the 1970s [17], [20], [38]. This paper follows the usual approach of exploiting parallelism by expanding multiple nodes of the search tree simultaneously. Its aim is to extend the existing literature of parallel branch and bound in two ways. First, it takes advantage of the power and special features of the CM-5 hardware: compared to most earlier multiple instruction multiple data (MIMD) [16] computing systems, the CM-5 processors can communicate with one another relatively rapidly and without great regard to the underlying network topology.

Second, and more importantly, this paper extends the current parallel branch-and-bound literature by describing an algorithm that is close to "industrial strength," bearing some resemblance to commercial serial mixed integer programming packages. The vast majority of existing work on parallel implicit enumeration has focused on specialized algorithms for "pure" combinatorial problems such as traveling salesman or vertex cover.

This paper is primarily a "proof of concept" demonstration — it attempts to show that one can indeed successfully exploit parallelism in problems with sufficiently large search trees. In-depth comparisons with possible alternative implementations are mostly reserved for a later, companion paper.

Section 2 of this paper lays out a succinct history of prior practical work in the field of parallel branch and bound. Section 3 briefly describes what a "semi-industrial-strength" general mixed integer branch-and-bound algorithm looks like, and §4 discusses issues of implementing such an algorithm in parallel. The remainder of the paper uses computational experiments to demonstrate that, on modern computing

* Received by the editors August 20, 1993; accepted for publication (in revised form) March 14, 1994.

† Mathematical Sciences Research Group, Thinking Machines Corporation, 245 First Street, Cambridge, Massachusetts 02142 (ecksteinᵃthink.com).

systems such as the CM-5, highly efficient use of tens to hundreds of processors to solve difficult, realistic mixed integer programming problems is a present-day reality. In addition, some of the experiments begin to tackle the issue of how one should centrally control the search process.

2. Prior work. The early literature of parallel branch and bound gives particular attention to the topic of *speedup anomalies.* Early on, it became evident that parallel adaptations of standard branch-and-bound techniques on p processors could frequently take less than $1/p$ of the time taken by their serial cousins [20], [26], [38]. More generally, it is possible for a run on $p_2 > p_1$ processors to consume less than p_1/p_2 as much time as one on p_1 processors. Such phenomena are known as *superlinear relative speedups* or *acceleration anomalies.* Fundamentally, they arise because the order in which subproblem nodes are expanded can, through the mechanism of pruning, affect the amount of work to be done later. Thus, the size and shape of the search tree can vary as the number of processors increases, an effect called a *search anomaly.*

In a serial context, it seems generally impossible to predict what search ordering will lead to a maximum amount of pruning. The usual concept of *absolute* speedup, which can never be superlinear (see, e.g., [7, Chap. 1]) is thus difficult to apply to general branch and bound because it is hard to define a single reasonable "best" algorithm for the one-processor case. In this paper, all speedups are relative.

There is also the possibility of *deceleration* anomalies, that is, that branch and bound may take more time when instantiated on $p_2 > p_1$ processors than on p_1, even if coordination overhead is negligible. Conditions under which acceleration and deceleration anomalies can occur are discussed in [26], [27], [30], [31].

A second prominent topic in the literature concerns central versus decentralized control of the search process. Many experiments and simulations [9], [17], [20], [37], [38], [41] have involved keeping the pool of "live" search nodes in shared lockable memory. On systems lacking shared memory, an essentially equivalent scheme is to distinguish a particular processor as a "master" or "hub" responsible for assigning subproblem expansions to the remaining "slave" or "worker" processors [13]. Both of these centralized schemes, as well as the related "shared resource" scheme given in [10], suffer from a lack of theoretical "scalability" (see, e.g., [22]); when using a very large number of processors to search a proportionally large tree, shared-resource bottlenecks could keep processors idle for large amounts of time, thus limiting speedups.

For distributed-memory systems, it has been more popular to design algorithms that lack any distinguished processors, giving each processor its own local pool of "live" subproblems. Proposals for such approaches date back as far as the work of [14]. The central issue then becomes that of *load balancing*; at the simplest level, one must make sure that some processors do not run out of work while others have a large backlog of tasks. If the tree to be searched is fixed, or if a pure depth-first search strategy is appropriate, then load balancing the *quantity* of work is the main concern [24], [36]. However, discrete optimization problems may generally also require balancing local problem pool *quality*, as measured by the bound associated with each unexpanded tree node. Ineffective quality balancing is thought to result in *search overhead*, that is, detrimental search anomalies that increase the size of the search tree as the number of processors grows. While fully decentralized algorithms lack a guaranteed bottleneck, it is possible that search overhead could limit their empirical scalability, depending on the application.

Karp and Zhang [21] have proposed and analyzed a very simple load distribution

scheme (see also [34] and [39]): each processor takes tasks from a local pool in the usual "best-first" order, but sends each newly generated subproblem to the pool of a randomly chosen processor. After initial assignment, subproblems never move between processors. Thus, the term "work scrambling" might be more appropriate to the procedure than "load balancing." This scrambling approach is easily implemented on the CM-5, where arbitrary pairs of processors may communicate asynchronously, rapidly, and independently; §5 gives preliminary results comparing such an approach to centralized control. However, many multiprocessor systems, particularly those of older design, require the programmer to be sparing with communication and conscious of the interconnection topology, as transferring data to other processors, especially distant ones, can be very costly. Thus, many authors have suggested methods in which subproblems can be transferred between (either neighboring or arbitrary) pairs of processors only when one processor completely runs out of work [11], [15], [23], and [35]. Others have suggested schemes in which (usually neighboring) processors periodically share information about their work pools, and exchange problems in an attempt to improve the work balance [1], [2], [11], [23], [36]. In a similar vein, schemes in [25], [32], [33], [45], and [46] are particularly noteworthy for trying to balance quality as well as quantity. Wha and Ma [47] describe an early related scheme for synchronously shifting subproblems around a specially constructed processor ring; a hardware-independent variant appears in [1] and [2].

The relative effectiveness of centralized and decentralized task scheduling depends on the time it takes to expand subproblems, the efficiency of communication, the computer network topology, the load distribution strategy, and the susceptibility of the problem instance to search anomalies. Comparisons of central and distributed control are included in [1], [2], [25], [35], and [40]. This paper provides additional comparisons, although more exhaustive testing is reserved for a follow-on article.

Most of the above references discuss "pure" combinatorial problems with specialized bounding procedures, such as traveling salesman, vertex cover, set covering, or graph partitioning. Work by Cannon and Hoffman [10] is less restrictive, considering general 0–1 pure integer programming and using a sophisticated bounding procedure, but on at most eight processors. Boehning, Butler, and Gillett [9] presented a general method with a linear programming-based bounding procedure, but problem sizes and numbers of processors were small, and speedups very modest to nonexistent. The unpublished thesis of Abdel-Rahman [1] contains tests of a simplex-based branch-and-bound procedure for pure integer linear programming. However, these tests seem confined to a single problem instance; most of the Abdel-Rahman analysis involves more specialized bound calculations. Starting from a general mixed-integer problem formulation, this paper presents parallel algorithms that are much closer to "industrial strength" than those of either [1] or [9], and tests them extensively with up to 128 processors on a set of problems derived from real applications.

3. Algorithm components. Consider a general mixed-integer program

$$
\begin{array}{lll}
\text{minimize} & c^\top x & \\
\text{subject to} & Ax \overset{\leq}{=} b & \\
& l \leq x \leq u & \\
& x_j \in \mathcal{Z} & \forall j \in \mathcal{I},
\end{array}
\tag{1}
$$

where $\mathcal{Z}$ denotes the integers, A is an $m \times n$ matrix, and $\mathcal{I}$ is a set of distinguished indices identifying the integer variables. The relation $\overset{\leq}{=}$ is meant to denote that some constraints are inequalities of either the $\leq$ or $\geq$ form, and some are equalities.

Components of the upper bound vector u may be $+\infty$ and components of l may be $-\infty$. Input is via the standard MPS file format. Let R denote the problem's continuous relaxation, that is, (1) without any integrality restrictions.

Most introductory operations research texts describe the fundamental branch-and-bound algorithm for (1); see, for example, [18, §18.4]. Basically, the method explores a binary tree of subproblems. At any time t, denote the current pool of "live" subproblems by $\mathcal{L}(t)$. These subproblems resemble R, except that the l and u vectors might be more restrictive. The method begins with the single "root" problem R, that is, $\mathcal{L}(0) = \{R\}$. At each step in the development of the search tree, the method removes a single problem P from the live pool and uses the simplex method to calculate the optimal value $z(P)$ of the associated linear program, along with a solution vector $x(P)$. If this vector does not meet the integrality restrictions, the method selects some noninteger *branching variable* $x_{j(P)}$, $j(P) \in \mathcal{I}$. It then creates an "up" child problem P^+ with $l_{j(P)} = \lceil x_{j(P)}(P) \rceil$, and a "down" child P^- with $u_{j(P)} = \lfloor x_{j(P)}(P) \rfloor$. It places these child problems, which otherwise resemble P, back in the pool. Problems whose bounds exceed some known feasible solution to (1) are "fathomed," that is, they are dropped from the pool.

The remainder of this section describes "industrial strength" refinements to this basic framework. For obvious competitive reasons, publishers of commercial mathematical programming packages rarely broadcast all the internal details of their codes; however, sufficient background information to construct a serviceable algorithm is contained in [4], [6], [28].

3.1. Branching rules. The key refinement over the "textbook" method concerns the selection of branching variables. First, the user may assign each integer variable x_j a *priority* Q_j. Furthermore, the method keeps track of *pseudocosts* for every integer variable. At any time t, let $\mathcal{T}(t)$ denote the tree of all subproblems P for which $z(P)$ is known. Then define

$$\mathcal{S}_j^+(t) = \left\{ P \in \mathcal{T}(t) \mid P^+ \in \mathcal{T}(t), j(P) = j \right\}.$$

The "up" pseudocost of variable x_j, $j \in \mathcal{I}$, at any time t such that $\mathcal{S}_j^+(t) \neq \emptyset$, is

$$\phi_j^+(t) = \left(\frac{1}{|\mathcal{S}_j^+(t)|} \right) \sum_{P \in \mathcal{S}_j^+(t)} \left(\frac{z(P^+) - z(P)}{\lceil x_j(P) \rceil - x_j(P)} \right).$$

This quantity attempts to measure how rapidly the subproblem optimal objective value increases, on average, as x_j is forced upward. When the set $\mathcal{S}_j^+(t)$ is empty, one must supply some default value for $\phi_j^+(t)$. A zero value would make branching on x_j appear to have no affect on the objective value. The algorithm tries to avoid branching on such variables (see (4)), so a zero default would tend to make it concentrate on a small set of branching variables and be reluctant to try new ones.

As a less "biased" alternative, the algorithm uses a global "typical" default value, computed via

$$\mathcal{S}^+(t) = \{ P \in \mathcal{T}(t) \mid P^+ \in \mathcal{T}(t) \}$$
$$\phi_j^+(t) = \left(\frac{1}{|\mathcal{S}^+(t)|} \right) \sum_{P \in \mathcal{S}^+(t)} \left(\frac{z(P^+) - z(P)}{\lceil x_{j(P)}(P) \rceil - x_{j(P)}(P)} \right).$$

At the very outset of the algorithm, $\mathcal{S}^+(t) = \emptyset$, in which case one takes $\phi_j^+(t) = 0$ for all j. So, in summary,

$$(2)\quad \phi_j^+(t) = \begin{cases} \left(\dfrac{1}{|\mathcal{S}_j^+(t)|}\right) \displaystyle\sum_{P \in \mathcal{S}_j^+(t)} \left(\dfrac{z(P^+) - z(P)}{\lceil x_j(P) \rceil - x_j(P)}\right) & \mathcal{S}_j^+(t) \neq \emptyset, \\ \left(\dfrac{1}{|\mathcal{S}^+(t)|}\right) \displaystyle\sum_{P \in \mathcal{S}^+(t)} \left(\dfrac{z(P^+) - z(P)}{\lceil x_{j(P)}(P) \rceil - x_{j(P)}(P)}\right) & \mathcal{S}_j^+(t) = \emptyset, \ \mathcal{S}^+(t) \neq \emptyset, \\ 0 & \text{otherwise.} \end{cases}$$

Note that, because branching variables are always noninteger, the denominators within the summations cannot be zero.

The "down" pseudocost $\phi_j^-(t)$, which attempts to measure the change in optimal objective as x_j is forced *downward*, is similarly defined by

$$\mathcal{S}^-(t) = \{P \in \mathcal{T}(t) \mid P^- \in \mathcal{T}(t)\},$$
$$\mathcal{S}_j^-(t) = \{P \in \mathcal{S}^-(t) \mid j(P) = j\},$$

$$(3)\quad \phi_j^-(t) = \begin{cases} \left(\dfrac{1}{|\mathcal{S}_j^-(t)|}\right) \displaystyle\sum_{P \in \mathcal{S}_j^-(t)} \left(\dfrac{z(P^-) - z(P)}{x_j(P) - \lfloor x_j(P) \rfloor}\right) & \mathcal{S}_j^-(t) \neq \emptyset, \\ \left(\dfrac{1}{|\mathcal{S}^-(t)|}\right) \displaystyle\sum_{P \in \mathcal{S}^-(t)} \left(\dfrac{z(P^-) - z(P)}{x_{j(P)}(P) - \lfloor x_{j(P)}(P) \rfloor}\right) & \mathcal{S}_j^-(t) = \emptyset, \ \mathcal{S}^-(t) \neq \emptyset, \\ 0 & \text{otherwise.} \end{cases}$$

When taking a branching decision for subproblem P, the algorithm calculates a "score" for each x_j, $j \in \mathcal{I}$ that is outside some fixed tolerance of being integer. To compute the score, one lets

$$d_j^+ = \lceil x_j(P) \rceil - x_j(P), \qquad d_j^- = x_j(P) - \lfloor x_j(P) \rfloor,$$

and calculates the estimated degradations of the objective

$$D_j^+ = \phi_j^+(t) d_j^+, \qquad D_j^- = \phi_j^-(t) d_j^- .$$

Then, the score is

$$(4)\quad \sigma_j = \alpha_0 Q_j + \alpha_1 \min\{d_j^+, d_j^-\} + \alpha_2 \min\{D_j^+, D_j^-\} + \alpha_3 \max\{D_j^+, D_j^-\},$$

where $\alpha_0, \ldots, \alpha_3$ are (typically positive) scalar parameters. The branching index $j(P)$ is the index maximizing the score, with ties broken arbitrarily. Usually, α_0 is very large, so that high-priority variables are given preference.

3.2. Search order. Subproblems are expanded in the usual best-first order, with an initial depth-first phase. Let $\overline{z}(t)$ denote the value of the *incumbent solution*, the best-known fully integer-feasible solution to (1) seen by time t; if no such solution is known, $\overline{z}(t) = +\infty$. Also, define $z(P,t)$ to be the lower bound $z(P)$, if it has already been computed. Otherwise, $z(P,t)$ is the corresponding bound for P's parent. For completeness, assume that the bound for the "parent" of R is $-\infty$. A lower bound on the optimal value of (1) is

$$\underline{z}(t) = \min\{\overline{z}(t), \min\{z(P,t) \mid P \in \mathcal{L}(t)\}\}.$$

The algorithm pursues a depth-first search of the subproblem tree as long as $\underline{z}(t)$ and $\overline{z}(t)$ differ by at least some fixed threshold percentage τ. Afterwards, it expands problems with minimal $z(P,t)$.

3.3. Incumbent heuristics. If a subproblem solution obeys all the integrality constraints, then it is clearly eligible for consideration as the incumbent. However, given an $x(P)$ that violates some integrality constraints, one may also attempt to modify it into a "nearby" candidate incumbent solution that *does* meet all these restrictions. Such procedures can produce aggressive pruning of the search tree without compromising the optimality of the final result. Usually, simple rounding of undesired fractional variables is unsatisfactory because it violates the linear constraints. A number of more sophisticated approaches have been proposed, such as the *pivot and complement* heuristic [5]. However, these methods generally require access to the innards of the simplex solver used to calculate $x(P)$.

Since such access was not available, the algorithm described here employs an (apparently) original alternative called a "shaking heuristic," which tries to use linear programming to round the solution, while still taking the linear constraints into account. The heuristic takes each fractional $x_j(P)$, $j \in \mathcal{I}$, restricts it to the range $[\lfloor x_j(P) \rfloor, \lceil x_j(P) \rceil]$, and gives it a very large perturbed cost coefficient intended to drive x_j to the closer of these bounds. One then solves the perturbed linear program. If all the integrality restrictions are then met, the resulting solution is a candidate incumbent. If not, all fractional variables have their perturbations' signs reversed, the perturbation magnitudes increase, and the process repeats. If no new incumbent is found, the heuristic halts after a fixed number of iterations.

The shaking heuristic is crude, but seemed adequate for the "proof of concept" purposes of this paper. Since it involves solving multiple linear programs, it is more time consuming than the lower bound calculation itself. Therefore, it is not run for every subproblem, but only for a randomly selected subset.

3.4. Other refinements. The algorithm also incorporates a standard *locking heuristic*: after solving P, one checks the linear programming (LP) reduced costs of all integer variables x_j that are at their bounds. If the absolute value of such a reduced cost exceeds $\overline{z}(t) - z(P)$, then x_j is fixed at its present value in all of P's descendents. This procedure is valid because any descendant solution with a different but still integral value of x_j would necessarily be fathomed.

The algorithm fathoms problems when either of the conditions

$$\overline{z}(t) - z(P,t) \leq \epsilon \qquad \text{or} \qquad \frac{\overline{z}(t) - z(P,t)}{\max\{|\overline{z}(t)|, |z(P,t)|, 1\}} \leq \bar{\epsilon}$$

hold. In the runs presented below, $\epsilon = 0$ and $\bar{\epsilon} = 10^{-7}$, that is, problems are solved to seven digits of precision. Increasing these parameters would allow the algorithm to solve problems more approximately.

4. Design of the parallel implementation. The CM-5 is an MIMD multiprocessor consisting of $p = 2^k$ processing nodes (PNs), each with a single SPARC–2 microprocessor and four optional vector floating point units. The basic philosophy of the implementation is not to attempt parallel solution of individual LP subproblems; instead, a standard serial simplex code produces the LP bounds. Each PN runs its own copy of the simplex method, and many PNs simultaneously expand different nodes of the search tree. The effort in adapting a high-quality sparse simplex solver to use the CM-5 vector units would have been substantial, and not likely to be

very rewarding due to the short vector lengths typically encountered in commercial mixed-integer applications. Therefore, no such adaptation was attempted and, for the purposes of this paper, it is sufficient to consider the CM-5 as a closely coupled set of SPARC microprocessors without global memory.

The processors of the CM-5 are interconnected by two user-accessible networks, the *data* network and the *control* network. The control network is dedicated to global computation and synchronization operations. The data network is designed to transmit message packets between pairs of processors, and has essentially a "fat tree" structure (see, e.g., [29]). This architecture makes it possible for many arbitrarily chosen pairs of nodes to exchange data packets without "blocking" one another. Routing latency is low: packets traverse the data network in microseconds, even in large configurations. In many instances, it is thus possible to program the CM-5 as if it were a densely connected set of processors in which the available raw bandwidth between any pair of processors is about five megabytes per second.

The branch-and-bound algorithm was implemented in the C language, using the CMMD 3.0 message passing programming environment [43] and the CMOST 7.2 operating system. CMMD is a "control parallel" programming environment, as distinguished from the "data parallel" environments that also exist for the CM-5. The main limitation in programming under CMOST/CMMD is the lack of any shared memory or conceptual global address space. This limitation is partially a function of the hardware and partly due to the current software.

CMMD 3.0 incorporates a useful facility called *active messaging* [44]. In CMMD, an active message is a single data network packet, the first four bytes of which contain the address of a function f, called the *handler*. Such messages behave like asynchronous remote procedure calls: when an active message arrives at a processor where user interrupts are enabled (the default), it immediately causes suspension of the current program and execution of the handler f, with the remainder of the message supplied to f as arguments. Once the handler has been evaluated, control returns to the suspended program. When interrupts are disabled, arriving active messages are queued, and execute when the receiver calls an explicit "polling" function. Active messaging has several desirable properties: it allows for many different kinds of messages without extensive conditional code to dispatch on the arriving message type; it does not require the elaborate, comparatively high-overhead request-to-send/acknowledge protocol needed to manage buffers for longer messages; and it permits a communications layer to operate in the "background," while "foreground" code (such as a standard simplex solver) may be completely oblivious to being used in a parallel environment.

4.1. Centralized versus distributed control. The key design decision was whether to use a centralized mechanism to control the search, with the attendant possibility of bottlenecks, or a fully decentralized scheme, with the risk of search overhead and load imbalance. Initially, a decision was taken to implement a centralized scheme, and then to assess the seriousness of the resulting bottlenecks. Therefore, one processor (PN number 0) was set aside to be the "hub" processor controlling the search, while the other processors are "workers." The workers primarily calculate LP subproblem bounds, using the standard simplex solver CPLEX 2.1. There was no attempt to use the mixed integer capabilities within CPLEX, except for the MPS file reading routine.

The hub allocates work by treating the pool of active subproblems as a last-in, first-out stack during the initial depth-first phase, and as a heap during the best-

first phase. In the best-first phase, the hub assigns idle workers to subproblems with minimal bounds.

Later versions of the code had the option of designating an arbitrary number h of processors to function as hubs. Each hub maintains its own pool of problems and controls a fixed set of about p/h workers. So far, the implementation contains only a variant of the simple Karp–Zhang work-scrambling scheme; when a worker creates a new subproblem, it reports the problem to a randomly chosen hub. However, only the appropriate controlling hub can assign a bound calculation to a worker. Each hub uses the same strategy as in the single-hub algorithm, but with respect to problems in its local pool. This implementation made possible some initial experiments into the merits of various levels of distributed versus central control. Note that making hubs out of some $\Omega(p)$-sized subset of processors should yield a method as theoretically scalable as a fully decentralized one, since each hub will be responsible for $O(1)$ workers.

4.2. Nature of the active pool. In some commercial branch-and-bound packages, the active pool consists of *solved* subproblems [28]: to expand a node, one must split it into two children, compute their LP bounds, and, if they survive fathoming, place them back in the pool. Other packages maintain instead a pool of *unsolved* problems [28]. In this case, a node expansion consists of computing the LP bound of a subproblem, and, again, if it survives fathoming, splitting it into children that then return to the pool. The bound value associated with an unsolved problem is that of its parent. Some packages may store a mixture of solved and unsolved nodes or even some partially solved subproblems [28].

The solved-pool method saves memory and takes branching decisions as late as possible, thus using the best possible pseudocost information. In a distributed-memory parallel context, however, some interesting tradeoffs arise. In the solved-pool approach, the basic work of node expansion involves bounding at least two subproblems, whereas only one bound per expansion is needed with an unsolved pool. As the nonbounding work to be done at each node is relatively small, the unsolved-pool approach has a significantly finer task granularity, leading to more potential parallelism.

On the other hand, the solved-pool method implies a lower communication burden, because subproblems whose solutions meet all integrality restrictions, or that are fathomed immediately upon solution, need never pass through the pool. In hub-and-worker implementations, this implies that such problems, which are leaf nodes of the fully developed search tree, need never be communicated to a hub. If the search tree has size T, it has $\lceil T/2 \rceil$ leaves, although some leaf nodes may have to be communicated because they might be fathomed well after their bound calculations.

One is thus faced with a choice between an approach that has approximately half the task granularity with one that has potentially twice the volume of communication. However, further consideration of programming details favors the unsolved method. The children of a given subproblem differ in only one respect, the bounds on one particular variable. Therefore, it is wasteful to communicate them to a hub as completely distinct data structures. Instead, one should send and store a single structure containing all the shared data, plus a "splitting tag" identifying the branching variable. Barring fathoming, each element in the pool must be dispatched twice, once for each child, before it can be deleted. This scheme reduces the number of extra messages sent by the solved-pool scheme by a factor of 25%, and the number of extra data elements passing through the active pool by 50%. These savings come at the cost of transmitting and storing marginally larger data structures.

CM-5 systems can potentially be scaled to many thousands of processors, and communication is relatively efficient. Thus, an implementation favoring smaller task granularity at the cost of communications volume would seem more desirable, arguing in favor of an unsolved pool. Considering the economies afforded by splitting tags, a decision was taken to use an unsolved pool. Previous work with hub-and-worker schemes has tended to use a solved-pool approach [13], which may be more appropriate for slower communications, relatively small numbers of processors, and more specialized problem domains. One prior experiment in parallel branch and bound for the TSP [37] incorporated *two* pools, one for solved problems and one for unsolved problems; both resided in shared memory.

4.3. Quasidistributing pool storage. In the first version of the code, a single hub processor stored the entire pool of live subproblems. Associated with each subproblem pair was a splitting tag, the LP relaxation value of the parent, modified bound information for the integer variables, and a brief symbolic description of the parent's optimal simplex basis. This last information is crucial for "warm starting" the simplex method. The pool was stored as a heap, organized by bound. This scheme proved serviceable for medium-sized problems, but increasing the number of variables, number of constraints, and size of the active pool uncovered serious *memory* load balancing problems. The pool began to consume prohibitive amounts of memory at the hub processor, while similarly large amounts of memory went unused at the worker processors.

Later versions of the code remedied this problem by using active messages to effectively implement distributed memory for the bulk of the subproblem data. Under this scheme, a worker creating a new subproblem pair leaves the associated variable bound, starting basis, and miscellaneous information in local memory. All it communicates to the hub is a small "token" specifying the parent's LP bound, the worker's processor number, and the worker local memory address of the remaining data. This minimal information is all that the hub processor needs to allocate subproblems to workers. The hub contains a "scale model" of the live pool, with the bulk of the data being distributed randomly among the worker processors; active pool storage is thus "quasidistributed."

Suppose that the hub learns that a particular worker, say worker B, is idle. According to the best-first rule, some subproblem P should be evaluated next. Chances are, however, that this problem is stored on a different worker, say worker A, which is most likely computing the bound of some other subproblem. To deliver the data needed to evaluate P, the hub sends an active message to A. This message interrupts A and instructs it to begin an asynchronous background transfer of P's data, including all bound and basis information, to B. Worker A then returns to its foreground task. When the transfer is complete, B begins work on the newly arrived problem, while A is again interrupted, and decrements problem P's count of unevaluated children. If this count reaches zero, A deletes P from its memory. The outlines of this quasidistributed memory scheme are depicted in Fig. 1.

When a problem known to the hub is fathomed, the hub not only deletes the corresponding token from its own memory, but also sends an active message to the corresponding worker. This message interrupts the worker and tells it to delete the problem from local memory as soon as any associated outstanding background transfers are complete. Usually, deletion is immediate.

In addition to balancing the memory load, the quasidistributed technique has the benefit that all communication involving the hub reduces to short, low-overhead,

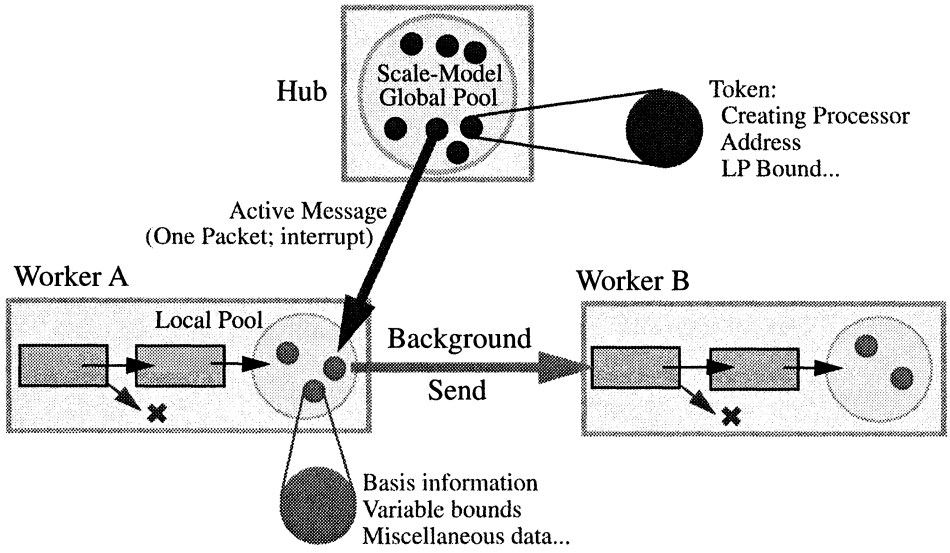


FIG. 1. *Implementation of quasidistributed subproblem storage.*

active messages, thus increasing the number of workers the hub can manage before bottlenecks ensue. More crucially, most communication does not pass through the hub. Instead, the vast majority of data is transmitted in relatively large batches between random pairs of worker processors. A number of such transfers may be taking place at any one time. This kind of random, possibly simultaneous point-to-point communication is well suited to the CM-5 data network architecture.

A similar scheme was proposed earlier and independently in [35] and [40]. The "transputer" systems used in that work have a less advanced communication architecture than the CM-5, but permit complex multithreaded user code at each processor. This multithreading capability, like user interrupts in CMMD, permits separate computation and data-transport layers to coexist on each processor, even when the computation layer relies heavily on standard serial software that does not contain explicit polling or context-switching features.

While the quasidistributed memory scheme has been described for the case of one hub, it functions equally well when there are multiple hubs.

4.4. Distribution of incumbent values. The value of the incumbent is critical to search-tree pruning at both the workers and hubs. Ideally, the incumbent value would exist in a global memory location that could be accessed with equal and minimal overhead by any processor. Unfortunately, the CM-5, like all current large-scale multiprocessors, lacks true global memory. In this case, however, active messages can provide a very workable emulation of a global incumbent register.

This active message scheme works as follows: each processor has a memory location called `incumbent`. Whenever a worker constructs a candidate incumbent, it compares the corresponding objective value to `incumbent`. If the candidate has a better value, the candidate solution is copied to a local buffer, its value replaces `incumbent`, and the worker initiates an asynchronous broadcast operation. If there are $p = 2^k$ total processors, the worker starts this broadcast by sending an active message carrying the new value to the k processors whose addresses differ from its

own in exactly one bit. Consider now a receiver of one of these messages, whose address differs from the sender at bit number $\ell \leq k$. When the message arrives, the receiver compares the contained value to its own `incumbent`. If `incumbent` is better, the receiver does nothing. If the new value is better, the receiver updates `incumbent` and forwards the message to the $\ell - 1$ processors whose addresses differ in lower-order bits than ℓ. These messages are then processed and/or forwarded by those processors, and so on. In this way, all processors learn the new incumbent value within $O(k) = O(\log p)$ time, or its distribution is stopped through collision with a better, simultaneously spreading value. The initial value of `incumbent` is $+\infty$.

Since this scheme operates under interrupts, foreground code merely sees a memory location that occasionally changes to reflect an improving global incumbent. On termination, the originating worker of the final value of `incumbent` has an optimal solution to (1) in its local buffer. In [10], a similar but apparently more centralized interrupt-driven method distributed incumbent values, although on a smaller scale.

4.5. Storage of the pseudocosts. Like the incumbent value, the pseudocost tables are global information. It appears that no prior parallel branch-and-bound implementation has attempted to maintain any global data of this sort. Production of new pseudocost information is a much more common event than creation of new incumbents, and the amount of data involved is much greater; hence global memory emulation seems less practical for pseudocosts than for the incumbent value.

Much as with the search control strategy, a decision was taken to start with a centralized approach, assess the severity of any resulting bottlenecks, and then move to a more distributed method if necessary. Thus, another distinguished processor, the *pseudocost server*, was set aside to store the pseudocost tables. In the current implementation, this server is the highest-numbered processor, with address $2^k - 1$.

Whenever a worker computes the bound of a subproblem, it sends an active message packet to the pseudocost server. This packet identifies the problem's branching variable, states whether the subproblem was an "up" or "down" child, and contains a single number to adjoin to one of the averaging formulas (2) or (3). When the message arrives, the server immediately updates the master pseudocost tables.

When a worker needs to make a branching decision, it compiles a list of the integrality-violating variables and sends it to the server. While the worker is waiting for a reply, it occupies itself with other tasks, such as the locking heuristic. When the server receives a list of variables, it collects the corresponding pseudocosts and returns them to the sender.

Admittedly, this centralized approach is somewhat crude, and the server will certainly become a bottleneck in various limiting cases. More scalable approaches might include having a number of pseudocost servers, each responsible for servicing $O(1)$ workers. These servers could periodically exchange information to prevent their tables from becoming overly discrepant. Pseudocosts are purely heuristic information, so there is no theoretical harm in using approximate or outdated versions of them. This paper does not test the viability of alternative pseudocost storage schemes; such comparisons will be addressed in follow-on research.

4.6. Starting and stopping. Starting the method is quite straightforward: one of the hub processors sends a description of R to one of its workers.

Detecting termination is trickier, and the possible existence of multiple hub processors complicates the situation considerably. One function of the CM-5 control network is to maintain a one-bit virtual "global or" register. Each processor contributes a bit to the register, which computes the inclusive or of the values contributed by all

processors, although with a few microseconds of possible latency. In this application, the global or register serves as a systemwide "not done" flag. The pseudocost server and all workers contribute a value of zero. Each hub processor contributes a value of zero if it has no problems in its local heap, and all the workers it controls are idle. Otherwise, it contributes a one. Thus, the register will contain the value one so long as there is work pending or in progress anywhere in the system. As soon as a processor reads a global value of zero from the register, it assumes termination has occurred, and enters the "clean-up" phase of the algorithm. Some care must be taken in the rare case that a worker reports a subproblem to a hub that has an empty pool and no busy workers, and then reports itself idle to a different hub, which then enters the same state. A simple "handshake" protocol between the worker and the first hub prevents spurious termination in this uncommon situation.

This final cleanup phase uses the reduction capabilities of the control network to compute the global minimum of the `incumbent` over all processors. Then, from among worker processors whose incumbent buffers correspond to this value, the system selects the one with the lowest network address. This step requires one additional control network reduction. The elected processor then outputs the optimal solution to a file.

5. Computational experiments. To test the branch-and-bound code, a set of test problems was assembled mostly from the MIPLIB [8], a public collection of models derived from real industrial applications. The test problems were selected to have large, but not astronomical, search trees, so that they would be tractable, but yet have some parallelism to exploit. To elaborate, problems with small search trees, which are quite common in the MIPLIB, do not offer much parallelism to this type of algorithm. The main challenge in these kinds of problems tends to lie in solving R. To efficiently apply parallel computing in such cases, one would have to parallelize the calculation of the LP bounds themselves. On the other hand, problems with truly immense search trees are likely to offer good speedups, yet remain insoluble in a reasonable time. Solving such problems by a branching methods requires some combination of reformulation with specialized and/or polyhedral techniques to strengthen lower bounds and locate good incumbents. However, such specialization may still be compatible with the eventual application of the kind of parallel search that is described here.

In general, test problems were selected to have search trees of roughly 1,000–100,000 search nodes, although even larger trees are probably tractable. The largest CM-5 configuration available for the tests had 128 processors. In the interest of limiting run times, problems with very difficult LP relaxations were excluded. Table 1 gives the characteristics of the test problems; most are in fact pure integer programs. It should be noted that for some individual problems in the test set, there may exist specialized algorithms that may outperform simple branch and bound; for example, see [19] for `air05`. No comparison with such methods is intended.

5.1. Runs using a single hub. To test the performance and limitations of centrally controlled search, the code was configured to use one hub, and run on the problems of Table 1 using varying numbers of processors. CM-5s with 32 and 64 processors were consistently available throughout the testing process and a 128-processor configuration was sometimes available. The code can be configured to leave some processors unused, so there were also simulated runs with 3, 4, 8, and 16 processors. Accounting for the hub and the pseudocost server, the number of worker processors is two less than the total number of processors. Thus, a three-processor run uses only one worker, and is essentially equivalent to a serial implementation, although there is

TABLE 1

Summary of test problems. All are from the MIPLIB, *except* `dcmulti`, *supplied by Jeremy Shapiro of the* M.I.T. *Sloan School of Management, and* `qiu`, *from Yuping Qiu of USWest Advanced Technologies.*

		Variables			
Problem	Constraints	Total	Binary	General Integer	Application
`air05`	426	771	771		Aircrew scheduling
`bell3a`	123	133	39	32	Fiber-optic network design
`dcmulti`	290	548	75		Multi-period facility location
`egout`	98	141	55		Drainage network design
`flugpl`	18	18		11	Air transport
`gen`	780	870	144	6	Electric generator scheduling
`khb05250`	101	1350	24		Capacitated plant location
`l152lav`	97	1989	1989		(undisclosed)
`lseu`	28	89	89		(undisclosed)
`mod010`	146	2655	2655		(undisclosed)
`p0033`	16	33	33		(undisclosed)
`p0201`	133	201	201		(undisclosed)
`p0282`	241	282	282		(undisclosed)
`p0291`	252	291	291		(undisclosed)
`pipex`	25	48	48		(undisclosed)
`qiu`	1192	840	48		Fiber-optic network design
`rgn`	24	180	100		(undisclosed)
`stein27`	118	27	27		(undisclosed)
`stein45`	331	45	45		(undisclosed)

a little communication overhead.

Message transmission times in the CM-5 data router are nondeterministic, so messages do not always arrive at the hub in the same order from run to run. Such effects can alter the search order pursued by the hub, leading to different sequences of incumbents and different evolution of the pseudocosts. In turn, these variations can lead to changes in pruning and branching variable choice, and thus to different search trees. Only in the three-processor case will successive runs search completely identical trees. To reduce the chance of random effects skewing the results, an attempt was made to execute at least 15 runs for each processor/problem combination. Runs of the more difficult problems on smaller configurations were time consuming, so fewer of them could fit in the available time. A few combinations, such as `air05` below 32 processors, had to be omitted altogether.

For all except the difficult problems `air05` and `qiu`, there was a single set of run time parameters, namely, $\alpha_1 = 1$, $\alpha_2 = 10$, and $\alpha_3 = 1$ in the scoring formula (4). The MIPLIB files do not provide branching priorities, so the Q_j were all zero, and the value of α_0 was irrelevant. The threshold for switchover from depth-first to best-first search was $\tau = 50\%$. CPLEX was configured to use primal simplex to solve the root problem, dual simplex for all subsequent subproblem bounds, and primal simplex for the shaking heuristic.

The problem `qiu` has extremely weak initial LP bounds, so its τ was increased to 100% to prevent a lengthy and fruitless prolongation of the depth-first phase. For `air05`, CPLEX was configured to use dual steepest-edge simplex in all contexts; otherwise, the parameters were not altered. All runs were under release 3.0.2 of CMMD and version 7.2β2P2 of the CMOST operating system. Apart from CPLEX and CMMD, the program consisted of some 3,900 lines of C code, compiled with the SPARC Gnu C Compiler, version 2.3.3, at optimization level `-O2`.

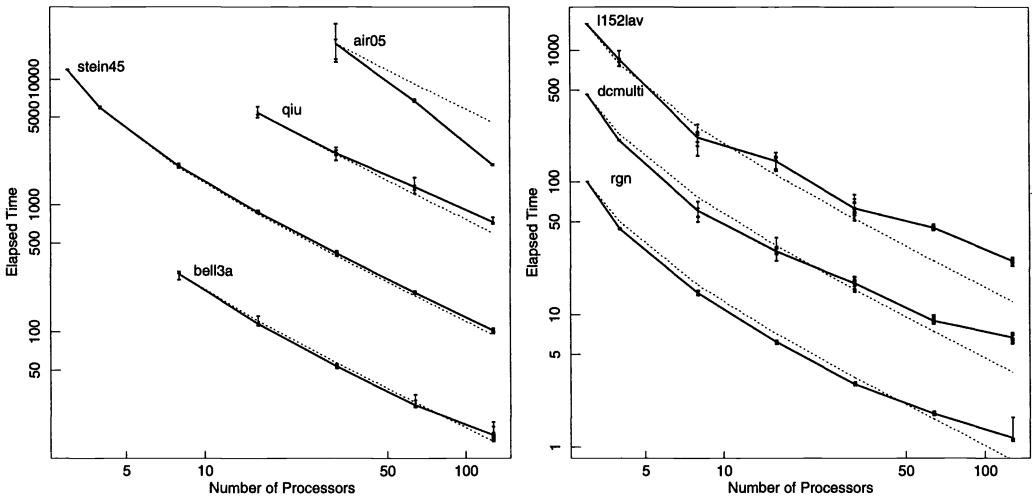


FIG. 2. *Detailed performance results for hard and medium-difficulty problems. Timings are in seconds.*

Figure 2 gives details of the code's performance, with the first graph displaying four of the most difficult problems and the second graph giving some more typical ones. Both axes are logarithmic; the horizontal axis gives the total number of processors p and the vertical axis gives total seconds of run time. Heavy lines connect the average run times for successive processor configurations, while the light, dotted lines show the "ideal" performance that would be expected if solution time scaled linearly with the number of worker processors. These latter lines curve for small p because the number of workers is two less than the total number of processors. Note that for the easier problems, there seems to be a "point of diminishing returns" after which the benefits of adding processors begin to disappear.

Table 2 gives a summary of the single-hub results for all the test problems. Here, $p_{\min}$ denotes the smallest number of processors used to solve a given problem, and $T_{p_{\min}}$ is the associated average size of the search tree. In general, let t_p denote average solution time on p processors. The next column in the table gives the estimated time $\widehat{t_1}$ needed to solve the problem on a single processor. If $p_{\min} = 3$, implying a single worker, $\widehat{t_1}$ is simply t_3. For larger $p_{\min}$, $\widehat{t_1}$ is estimated by assuming that a one-processor run would also search a tree of size $T_{p_{\min}}$.

Define the average relative speedup s_p on p processors to be $\widehat{t_1}/t_p$, along with the corresponding average relative efficiency $e_p = s_p/p$. The next group of columns in the table gives performance statistics t_{64}, s_{64}, and e_{64} for the 64-processor configuration. Finally, let p^* denote the largest tested processor configuration for which $t_p \leq (0.6)t_{p/2}$; it attempts to objectively estimate an "appropriate" configuration for each problem. The last section of the table gives p^* and the associated performance statistics t_{p^*}, s_{p^*}, and e_{p^*}.

Note that only for a few problems does p^* fall much below $T_{p_{\min}}/100$; that is, the benefits of parallelism tend to persist at least until there are about 100 serial search tree nodes for every processor. When attempting to apply even more processors, a number of effects mitigate against continued speedup. In all but the hardest problems, there was some *starvation*, meaning that workers were idle because no work was available to them. Any parallel search algorithm can encounter this effect when

TABLE 2
Summary of single-hub computational results, sorted in order of problem difficulty.

Problem	$p_{\min}$	$T_{p_{\min}}$	$\widehat{t}_1$	t_{64}	s_{64}	e_{64}	p^*	t_{p^*}	s_{p^*}	e_{p^*}
air05	32	4,891	6.6 days	1.9 hr.	84	131%	128	34.9 min.	272	212%
qiu	16	28,203	20.9 hr.	23.4 min.	54	84%	128	12.2 min.	103	80%
stein45	3	109,125	3.3 hr.	3.4 min.	58	91%	128	1.7 min.	115	90%
gen	16	3,655	1.2 hr.	2.9 min.	24	37%	64	2.9 min.	24	37%
bell3a	8	43,817	28.1 min.	26.3 sec.	64	100%	128	15.2 sec.	111	86%
l152lav	3	2,623	26.4 min.	45.0 sec.	35	55%	128	25.5 sec.	62	48%
p0282	3	9,894	20.3 min.	28.0 sec.	43	68%	128	15.1 sec.	81	63%
mod010	3	1,181	9.7 min.	28.6 sec.	20	32%	16	28.1 sec.	21	129%
dcmulti	3	3,671	7.7 min.	9.0 sec.	51	80%	64	9.0 sec.	51	80%
khb05250	3	3,583	5.9 min.	7.4 sec.	48	75%	64	7.4 sec.	48	75%
stein27	3	9,479	5.3 min.	5.9 sec.	55	86%	128	3.4 sec.	95	74%
p0201	3	2,301	4.9 min.	7.2 sec.	41	64%	32	10.9 sec.	27	84%
flugpl	3	18,443	3.3 min.	4.1 sec.	48	75%	16	5.6 sec.	36	222%
egout	3	7,207	3.3 min.	6.3 sec.	31	48%	64	6.3 sec.	31	48%
lseu	8	6,848	2.3 min.	3.5 sec.	39	61%	64	3.5 sec.	39	61%
rgn	3	3,705	1.6 min.	1.8 sec.	55	86%	64	1.8 sec.	55	86%
pipex	3	5,234	1.6 min.	1.0 sec.	97	152%	32	1.5 sec.	65	204%
p0291	3	1,055	1.3 min.	3.3 sec.	24	37%	8	8.1 sec.	10	121%
p0033	3	2,357	33.2 sec.	1.4 sec.	23	36%	16	2.2 sec.	15	92%

the number of processors is large in relation to the search space. Starvation was particularly noticeable in the large-configuration runs of mod010 and p0291, which have relatively small trees. Another common effect, more peculiar to branch and bound, can be called *inflation*: at some point, the search tree starts to grow with the number of processors, because extra available workers allow the expansion of subproblems that would otherwise soon be fathomed. Processors thus do unrewarding work instead of simply sitting idle. Inflation was pronounced for the larger runs of p0291 and p0033.

Two further impediments to indefinite speedup are hub and pseudocost server overload: if these special processors become too busy, workers accrue idle time waiting for new work or for pseudocost data. Only in runs above 16 processors for the two models with the easiest subproblems, p0033 and flugpl, were such effects significant. It seems that a single hub is adequate up to about 3,000 subproblems per second, beyond which queueing effects start to take over. The p0033 search tree is fairly small, so runs beyond 16 processors are of little interest. Flugpl, however, has a search tree in the 10,000–20,000 node range, so one might expect it to be amenable to hundreds of processors. Unfortunately, hub and pseudocost server congestion, coupled with some inflation, prevent any major speedups beyond 16 processors. All the other problems in the test set seem well suited to a central control strategy in the studied range of CM-5 configurations.

Most of the e_{p^*} measures in Table 2 are in the 60–130% range. Notable exceptions are air05, gen, flugpl, egout, and pipex.

Gen exhibits a marked deceleration anomaly between 16 and 32 processors, and egout takes almost as long on four processors as on three. In both cases, an increase in search tree size is to blame. On the other had, flugpl and pipex show rather dramatic acceleration anomalies, due to shrinking trees, for small numbers of processors. Figure 3 shows these anomalous problems, along with p0282. Note that search tree size can be quite variable even among runs on the same number of processors, suggesting that these problems have intrinsically "volatile" search patterns. Figure 4 shows search

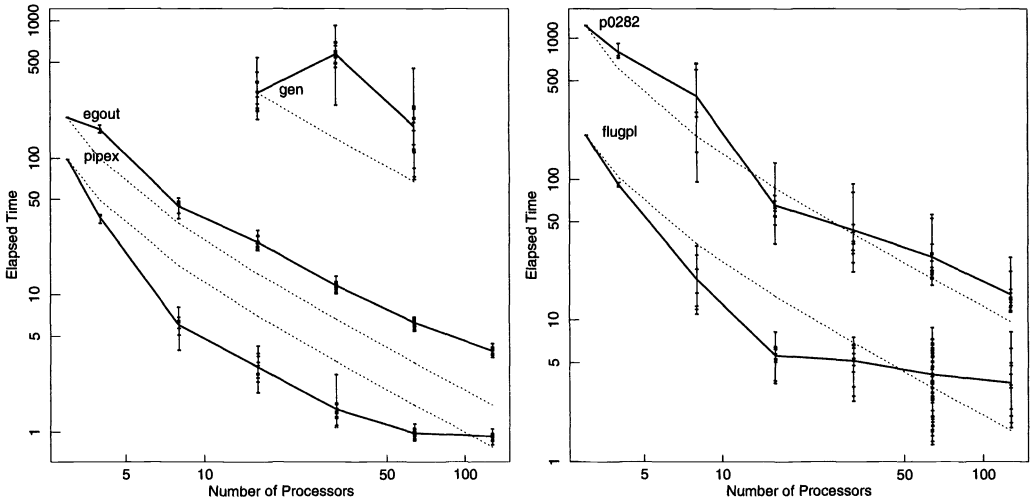


FIG. 3. *Detailed performance results for problems with large search anomalies and volatile search patterns.*

tree sizes for the problems of Figs. 2 and 3.

`Air05` is rather curious. It shows strong superlinear speedup, but its search trees do not change size appreciably. Since better incumbent information is available sooner in the more parallel runs, the locking heuristic causes a greater proportion of subproblems to become infeasible. Also, the code uses a CPLEX feature that directs subproblems to be aborted if they provably exceed some "cutoff" value (in this case, `incumbent`). Aborted and infeasible problems take less time, hence the extra speedup.

5.2. Simulation of a more decentralized implementation. To test the practicality of more fully decentralized approaches, the code was then configured so that the number of hubs was equal to the number of workers. Each hub is thus responsible for exactly one worker. This is not a very efficient way to operate the code in practice, but it allows each hub-worker pair to effectively simulate a single processor of a fully decentralized implementation with Karp–Zhang work scrambling. The code was configured to have 62 hubs and 62 workers, and was run on a 128-processor configuration. Table 3 shows the results, in comparison to single-hub runs on a 64-processor system. The runs should be comparable because there are 62 workers in both cases. Recoding the implementation to merge the worker and hub functions should make similar decentralized results attainable using only half as many actual processors. Because the implementation still has quasidistributed pool storage, the resulting version of Karp–Zhang has the interesting property that starting basis and variable bound data are only transmitted between processors for search nodes that are actually expanded. However, since each hub controls only one worker, both members of a subproblem pair are always evaluated on the same worker. Thus, the granularity advantages of using an unsolved pool are lost, and basis and bound information are transmitted twice, when only once would suffice.

As might be expected, the decentralized approach is almost always slower than the centralized one; on average, the decentralized code run times are about 30% longer. The reasons why are fairly surprising. Search overhead is not the culprit; unexpectedly, search trees are typically *smaller* in the decentralized code. It appears

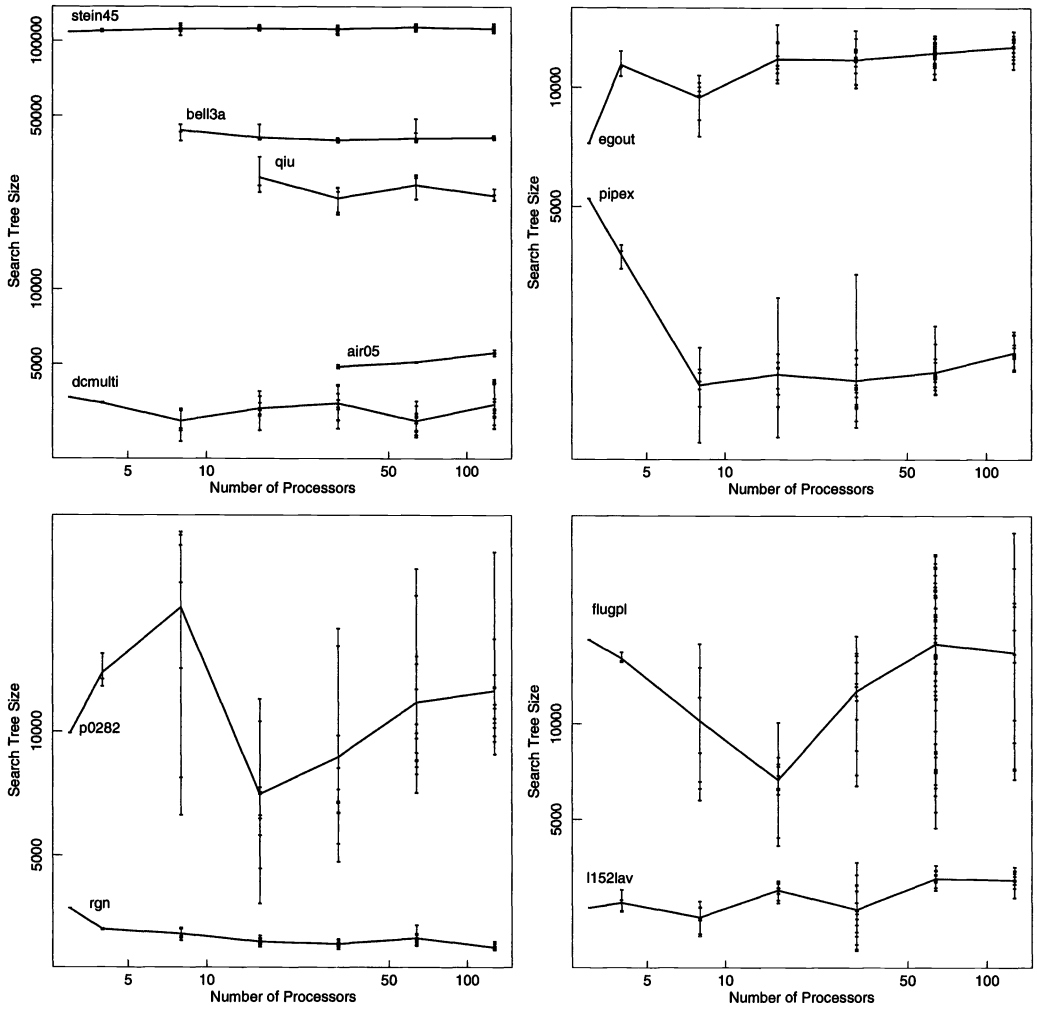


FIG. 4. *Search tree sizes for selected problems.*

that a certain amount of randomness in the search procedure can be beneficial; there may be implications here even for completely serial discrete search algorithms.

The main reason that the decentralized version is slower seems to be that high levels of starvation force the worker processors to be idle much of the time. It appears, then, that Karp–Zhang does a fairly good job of balancing work quality, but is less effective at evening out its quantity. Granularity loss may also be partially to blame, but it is hard to imagine how such an effect alone could multiply starvation by a factor of four, as in `khb05250` and `p0201`. The results suggest that the decentralized Karp–Zhang approach, when combined with a fairly simple load-balancing procedure, might be very effective in practice. If necessary, task granularity could be restored, at the cost of some communication, by splitting subproblem pairs upon creation and reporting to different random processors.

6. Conclusions. The main thrust of the work presented here has been to take advantage of the features of the CM-5 to build a branch-and-bound implementation

TABLE 3

Comparison of centralized and fully decentralized work distribution for 62 workers. The "% Starve time" column measures the fraction of the time that a hub had idle workers, but no work to assign. It is averaged over all hubs. `Air05`, `gen`, `lseu`, *and* `qiu` *were omitted to conserve computing resources.*

Problem	Average run time (seconds)			Average tree size			% Hub busy		% Worker busy		% Starve time	
	Hubs			Hubs			Hubs		Hubs		Hubs	
	1	62	Δ%	1	62	Δ%	1	62	1	62	1	62
`bell3a`	26.3	33.8	+28	40,528	40,745	+1	15	1.7	95	76	3	16
`dcmulti`	9.0	14.8	+65	2,952	3,469	+18	3	0.0	69	45	35	55
`egout`	6.3	8.3	+32	12,039	11,037	−8	20	0.3	90	62	12	37
`flugpl`	3.4	3.8	+11	15,194	10,608	−30	32	3.4	78	54	11	33
`khb05250`	7.4	9.2	+24	3,937	2,973	−25	5	0.1	91	57	10	43
`l152lav`	45.0	47.1	+5	3,240	2,844	−12	1	0.0	78	48	25	52
`mod010`	28.6	31.4	+10	0,712	0,554	−22	1	0.0	29	15	77	85
`p0033`	1.4	2.3	+57	4,498	5,191	+15	26	0.3	70	51	29	47
`p0201`	7.2	9.7	+35	2,713	1,781	−34	4	0.0	84	38	18	62
`p0282`	28.0	21.4	−24	11,781	7,931	−33	7	0.1	97	82	3	18
`p0291`	3.3	5.0	+49	1,654	1,454	−12	5	0.1	72	43	35	57
`pipex`	1.0	1.6	+65	1,919	2,247	+17	16	0.2	69	52	35	46
`rgn`	1.8	2.6	+45	3,130	2,908	−7	14	0.1	76	50	27	50
`stein27`	5.9	7.8	+33	9,543	9,635	+1	14	0.2	91	68	9	31
`stein45`	204.2	219.0	+7	114,014	113,788	+0	7	0.1	99	91	1	9
Average:			+29.6			−8.8	11	0.4	79	55	22	43

that performs well when the search process is centrally controlled. Furthermore, the effort involved a fairly realistic general mixed integer programming algorithm, and not a more "academic" method specialized to some narrow combinatorial problem class, although these too may have their place in some real applications.

Generally, the results have been very encouraging. For most of the test problems, drawn from a variety of disparate real-world models, either good speedups were seen with as many as 64 or 128 processors, run times were reduced to the order of seconds, or both. As one might expect, high processor efficiencies seemed easiest to obtain on the most difficult problems, provided that their search trees were reasonably large. The problem `air05`, which would take roughly a week to solve on a SPARC–2 workstation, can be solved in about an half an hour on a 128-processor configuration. The same system solves `qiu`, which would take about a day serially, in about twelve minutes. On easier problems, large speedups were harder to obtain, but solution times on the order of seconds could generally be achieved.

With the help of active messaging, it appears that a global address space or shared memory are not necessary for an efficient parallel branch-and-bound code. However, the implementation made heavy use of the speed and general purpose topology of the CM-5 data network; some of its techniques might not transfer well to systems with higher message latencies, lower bandwidth, or more constraining interprocessor connection patterns. For example, almost all subproblems are moved from one processor to another before solution, which would be generally inadvisable on an ethernet-connected cluster of workstations.

Experiments with the variably decentralized multiple-hub version produced some intriguing preliminary data. Work scrambling of the sort first suggested by Karp and Zhang seems to be quite effective in limiting search overhead without the need for central control. It is also easy to implement on machines, like the CM-5, that do not

impose large penalties for communication with "distant" processors. However, there appear to be practical difficulties in keeping processors busy, suggesting that some sort of load balancing layer needs to be added to make the approach practical.

The results suggest a wealth of future computational projects, most immediately in combining scrambling with some form of load balancing, and subsequently experimenting with various levels of centralization of the search process. Experiments with decentralizing pseudo-cost storage are also in order. Possibly, more generalizable results might be obtained by replacing or improving the shaking heuristic, one of the least sophisticated parts of the test code.

Nonetheless, the results make it clear that parallel computing is ready to handle a broad range of difficult, real-world mixed integer programming problems. The time thus seems ripe to incorporate the technology, possibly in more specialized form, into a functional decision support system. The cost effectiveness of such a system would of course depend heavily on the monetary value, time criticality, and computational properties of the application.

One should also consider strengthening parallel branch-and-bound procedures with cutting planes, yielding parallel branch-and-cut methods. Indeed, a highly specialized parallel branch-and-cut procedure running on a large workstation network has already successfully solved a very large instance of the traveling salesman problem [3], and Cannon and Hoffman [10] have explored more general purpose branch-and-cut methods on a small workstation cluster. Cuts represent potentially shared data and their handling presents potentially tricky issues akin to those arising around search control and pseudocost storage. The area certainly seems worth exploring, for parallelism and cutting planes are both clearly potent techniques in accelerating discrete numerical optimization. Their combination may yield even greater advances.

Acknowledgments. I would like to thank the following people for helping me with this project. Their help ranged from substantial computer coding to a few simple words of advice. At Thinking Machines: Doug Wiedemann, Xiru Zhang, Adam "Moose" Greenberg, Alan Mainwaring, and Bradley Kuszmaul. At CPLEX Optimization: Todd Lowe, Mary Finelon, Bob Bixby, and Irv Lustig. Elsewhere: Jerry Shapiro (M.I.T.), Michael Saunders (Stanford), Yuping Qiu (USWest), and Andy Boyd (Texas A&M). The editor, Robert Meyer, provided some helpful rewriting suggestions.

REFERENCES

[1] T. S. ABDEL-RAHMAN, *Parallel Processing of Best-First Branch and Bound Algorithms on Distributed Memory Multiprocessors*, Ph.D. thesis, Department of Computer Science and Engineering, University of Michigan, Ann Arbor, MI, 1989.

[2] T. S. ABDEL-RAHMAN AND T. N. MUDGE, *Parallel branch and bound algorithms on hypercube multiprocessors*, in Proceedings of the Third Conference on Hypercube Concurrent Computers and Applications, Pasadena, CA, 1988.

[3] D. APPLEGATE, V. CHVÁTAL, W. COOK, AND R. E. BIXBY, presentation at the DIMACS Workshop on Future Directions in Parallel Optimization, Piscataway, NJ, 1993.

[4] E. BALAS AND M. GUIGNARD, *Report on the session on branch and bound/implicit enumeration*, in Discrete Optimization II, P. L. Hammer, E. L. Johnson, and B. H. Korte, eds., Annals of Discrete Optimization 5, North-Holland, Amsterdam, 1979.

[5] E. BALAS AND C. H. MARTIN, *Pivot and complement — a heuristic for* 0-1 *programming*, Manag. Sci., 26 (1980), pp. 86–96.

[6] E. M. L. BEALE, *Branch and bound methods for mathematical programming systems*, in Discrete Optimization II, P. L. Hammer, E. L. Johnson, and B. H. Korte, eds., Annals of Discrete Mathematics 5, North-Holland, Amsterdam, 1979.

[7] D. P. BERTSEKAS AND J. TSITSIKLIS, *Parallel and Distributed Computation: Numerical Methods*, Prentice-Hall, Englewood Cliffs, NJ, 1989.

[8] R. E. BIXBY, E. A. BOYD, AND R. R. INDOVINA, *MIPLIB: a test set of real-world mixed-integer programming problems*, SIAM News, 25 (1992), p. 16.

[9] R. L. BOEHNING, R. M. BUTLER, AND B. E. GILLETT, *A parallel integer linear programming algorithm*, Europ. J. Oper. Res., 34 (1988), pp. 393–398.

[10] T. L. CANNON AND K. L. HOFFMAN, *Large-scale 0-1 programming on distributed workstations*, Ann. Oper. Res., 22 (1990), pp. 181–217.

[11] J. CLAUSEN AND J. LARSSON TRÄFF, *Implementation of parallel branch-and-bound algorithms — experiences with the graph partitioning problem*, Ann. Oper. Res., 33 (1991), pp. 331–349.

[12] CPLEX OPTIMIZATION INC., *Using the CPLEX Callable Library and CPLEX Mixed Integer Library*, Version 2.1, Incline Village, NV, 1993.

[13] A. DE BRUIN, A. H. G. RINNOOY KAN, AND H. W. J. M. TRIENEKENS, *A simulation tool for the performance evaluation of parallel branch and bound algorithms*, Math. Programming, 42 (1988), pp. 245–271.

[14] O. I. EL-DESSOUKI AND W. H. HUEN, *Distributed enumeration on between computers*, IEEE Trans. Comput., C-29 (1980), pp. 818–825.

[15] R. FINKEL AND U. MANBER, DIB — *a distributed implementation of backtracking*, ACM Trans. Programming Lang. Sys., 9 (1987), pp. 235–256.

[16] M. J. FLYNN, *Some computer organizations and their effectiveness*, IEEE Trans. Comp., C-21 (1972), pp. 948–960.

[17] S. H. FULLER, J. K. OUSTERHOUT, L. RASKIN, P. I. RUBINFELD, P. J. SINDHU, AND R. J. SWAN, *Multi-microprocessors: an overview and working examples*, Proc. IEEE, 66 (1978), pp. 216–228.

[18] F. S. HILLIER AND G. J. LIEBERMAN, *Introduction to Operations Research*, Holden-Day, San Francisco, CA, 1980.

[19] K. L. HOFFMAN AND M. PADBERG, *Solving airline crew scheduling problems by branch-and-cut*, Manag. Sci., 39 (1993), pp. 657–682.

[20] M. IMAI, T. FUKUMURA, AND Y. YOSHIDA, *A parallelized branch-and-bound algorithm: implementation and efficiency*, Sys. Comput. Controls, 10 (1979), pp. 62–70.

[21] R. M. KARP AND Y. ZHANG, *A randomized parallel branch-and-bound procedure*, Proc. ACM Sympos. Theory Comput., 20 (1988), pp. 290–300.

[22] V. KUMAR AND A. GUPTA, *Analyzing scalability of parallel algorithms and architectures*, Tech. Report TR 91-18, Department of Computer Science, University of Minnesota, Minneapolis, MN, 1991.

[23] V. KUMAR, G. Y. ANANTH, AND V. NAGESHWARA RAO, *Scalable load balancing techniques for parallel computers*, Tech. Report TR 91-55, Department of Computer Science, University of Minnesota, Minneapolis, MN, 1991.

[24] V. KUMAR AND V. NAGESHWARA RAO, *Parallel depth-first search. Part* II. *Analysis*, Internat. J. Parallel Programming, 16 (1987), pp. 501–519.

[25] V. KUMAR, K. RAMESH, AND V. NAGESHWARA RAO, *Parallel best-first search of state-space graphs: a summary of results*, in Proceedings of the AAAI-88 Seventh National Conference on Artificial Intelligence, St. Paul, MN, 1988.

[26] T.-H. LAI AND S. SAHNI, *Anomalies in parallel branch-and-bound algorithms*, Comm. Assn. Comput. Mach., 27 (1984), pp. 594–602.

[27] T.-H. LAI AND A. SPRAGUE, *Performance of parallel branch-and-bound algorithms*, IEEE Trans. Comput., C-34 (1985), pp. 962–964.

[28] A. LAND AND S. POWELL, *Computer codes for problems of integer programming*, in Discrete Optimization II, P. L. Hammer, E. L. Johnson, and B. H. Korte, eds., Annals of Discrete Mathematics 5, North-Holland, Amsterdam, 1979.

[29] F. T. LEIGHTON, *Introduction to Parallel Algorithms and Architectures: Arrays, Trees, and Hypercubes*, Morgan Kaufmann, San Mateo, CA, 1991.

[30] G.-J. LI AND B. W. WAH, *Coping with anomalies in parallel branch-and-bound algorithms*, IEEE Trans. Comput., C-35 (1986), pp. 568–573.

[31] ———, *Computational efficiency of parallel combinatorial* OR*-tree searches*, IEEE Trans. Software Engrg., 18 (1990), pp. 13–31.

[32] R. LÜLING AND B. MONIEN, *Two strategies for solving the vertex cover problem on a transputer network*, in Proceedings of the Third International Workshop on Distributed Algorithms, Nice, 1989.

[33] ———, *Load balancing for distributed branch and bound algorithms*, in Proceedings of the International Parallel Processing Symposium, Beverly Hills, CA, 1992.

[34] G. MANZINI AND A. M. SOMALVICO, *Probabilistic performance analysis of heuristic search using parallel hash tables*, in Proceedings of the International Symposium on Artificial Intelligence and Mathematics, Ft. Lauderdale, FL, 1990.

[35] G. P. MCKEOWN, V. J. RAYWARD-SMITH, S. A. RUSH, AND H. J. TURPIN, *Using a transputer network to solve branch and bound problems*, in Transputing '91, P. Welch, D. Stiles, T. L. Kunii, and A. Bakkers, eds., IOS Press, Amsterdam, 1991.

[36] V. NAGESHWARA RAO AND V. KUMAR, *Parallel depth-first search. Part* I. *Implementation*, Internat. J. Parallel Programming, 16 (1987), pp. 479–499.

[37] J. F. PEKNY AND D. L. MILLER, *A parallel branch and bound algorithm for solving large asymmetric traveling salesman problems*, Math. Programming, 55 (1992), pp. 17–33.

[38] E. A. PRUUL, G. L. NEMHAUSER, AND R. A. RUSHMEIER, *Branch-and-bound and parallel computation: a historical note*, Oper. Res. Let., 7 (1988), pp. 65–69.

[39] A. RANADE, *A Simpler Analysis of the Karp–Zhang Parallel Branch-and-Bound Method*, Report UCB/CSD 90/586, Computer Science Division, University of California, Berkeley, CA, 1990.

[40] V. J. RAYWARD-SMITH, S. A. RUSH, AND G. P. MCKEOWN, *Efficiency considerations in the implementation of parallel branch-and-bound*, Ann. Oper. Res., 43 (1993), pp. 123–145.

[41] C. ROUCAIROL, *A parallel branch and bound algorithm for the quadratic assignment problem*, Discrete Appl. Math., 18 (1987), pp. 211–225.

[42] THINKING MACHINES CORPORATION, *The Connection Machine* CM-5 *Technical Summary*, Cambridge, MA, 1992.

[43] ———, CMMD *Reference Manual*, Version 3.0, Cambridge, MA, 1993.

[44] T. VON EIKEN, D. E. CULLER, S. C. GOLDSTEIN, AND K. E. SCHAUSER, *Active messages: a mechanism for integrated communication and computation*, in Proceedings of the Nineteenth International Symposium on Computer Architecture, Gold Coast, Australia, 1992.

[45] O. VORNBERGER, *Implementing branch-and-bound in a ring of processors*, in Proceedings of the CONPAR 86: Conference on Algorithms and Hardware for Parallel Processing, Aachen, 1986.

[46] ———, *Load balancing on a network of transputers*, in Proceedings of the Second Annual Workshop on Distributed Algorithms, Amsterdam, 1987.

[47] B. W. WAH AND Y. W. E. MA, *MANIP — a multicomputer architecture for solving combinatorial extremum-search problems*, IEEE Trans. Comput., C-33 (1984), pp. 377–390.

SIAM J. OPTIMIZATION
Vol. 4, No. 4, pp. 815–832, November 1994
© 1994 Society for Industrial and Applied Mathematics
008

PARALLEL VARIABLE DISTRIBUTION*

M. C. FERRIS† AND O. L. MANGASARIAN†

Abstract. An approach is presented here for solving optimization problems in which the variables are distributed among p processors. Each processor has primary responsibility for updating its own block of variables in parallel while allowing the remaining variables to change in a restricted fashion, e.g., along a steepest descent, quasi-Newton, or any arbitrary direction. This "forget-me-not" approach is a distinctive feature of the algorithm that has not been analyzed before. The parallelization step is followed by a fast synchronization step wherein the affine hull of the points computed by the parallel processors and the current point is searched for an optimal point. Convergence to a stationary point under continuous differentiability is established for the unconstrained case, as well as a linear convergence rate under the additional assumption of a Lipschitzian gradient and strong convexity. For problems constrained to lie in the Cartesian product of closed convex sets, convergence is established to a point satisfying a necessary optimality condition under Lipschitz continuous differentiability of the objective function. For problems with more general constraints, convergence is established under stronger conditions. Encouraging computational results on the Thinking Machines CM-5 multiprocessor are given on a subset of the publicly available CUTE set of nonlinear programming test problems.

Key words. parallel optimization, nonlinear programming

AMS subject classifications. 90C30, 49D27

1. Introduction.

We present an approach for solving constrained optimization problems in which the variables are distributed among processors even when the original problem is not separable. We consider the general nonlinear program

$$\min_{x \in X} f(x), \tag{1.1}$$

where X is a nonempty closed convex set in the n-dimensional real space R^n and $f: \mathrm{R}^n \to \mathrm{R}$ has continuous first partial derivatives on R^n. Our objective in this paper is to distribute p blocks $x_1, \ldots, x_p$ of the variable x, where $x_l \in \mathrm{R}^{n_l}$, $\sum_{l=1}^{p} n_l = n$, among p processors. Given an iterate $x^i \in \mathrm{R}^n$, processor l has primary responsibility for updating block $x_l^i \in \mathrm{R}^{n_l}$ of the iterate x^i by solving the following problem.

Parallelization.

$$\begin{aligned} (y_l^i, \lambda_{\bar{l}}^i) &\in \arg\min_{x_l, \lambda_{\bar{l}}} f(x_l, x_{\bar{l}}^i + D_{\bar{l}}^i \lambda_{\bar{l}}) \\ &\quad \text{such that } (x_l, x_{\bar{l}}^i + D_{\bar{l}}^i \lambda_{\bar{l}}) \in X, \\ x^{i_l} &:= (y_l^i, x_{\bar{l}}^i + D_{\bar{l}}^i \lambda_{\bar{l}}^i). \end{aligned} \tag{1.2}$$

Here $\bar{l}$ denotes the complement of l in $\{1, \ldots, p\}$, $\lambda_{\bar{l}} \in \mathrm{R}^{p-1}$. For simplicity of notation, throughout the paper we allow the ordering of the variables to be changed, thus enabling us to write the variables that are handled by the lth processor first. The matrix $D_{\bar{l}}^i$ is an $n_{\bar{l}}$-by-$(p-1)$ matrix. It is formed by taking an arbitrary direction $d^i \in \mathrm{R}^n$, breaking it into blocks of $d_s^i \in \mathrm{R}^{n_s}$, $s = 1, \ldots, p$, consistent with the

* Received by the editors September 10, 1993; accepted for publication (in revised form) February 17, 1994. This material is based on research supported by Air Force Office of Scientific Research grant AFOSR-89-0410 and National Science Foundation grants CCR-9157632, CCR-9101801, and CDA-9024618.

† Computer Sciences Department, University of Wisconsin, 1210 West Dayton Street, Madison, Wisconsin 53706 (ferris@cs.wisc.edu, olvi@cs.wisc.edu).

distribution of the variables, and placing these vectors along the block diagonal of $D_{\bar{l}}^i$ as follows:

$$D_{\bar{l}}^i := \begin{bmatrix} d_1^i & & & & & & & \\ & d_2^i & & & & & & \\ & & \ddots & & & & & \\ & & & d_{l-1}^i & & & & \\ & & & & d_{l+1}^i & & & \\ & & & & & d_{l+2}^i & & \\ & & & & & & \ddots & \\ & & & & & & & d_p^i \end{bmatrix}. \tag{1.3}$$

Once the points x^{i_l}, $l = 1, \ldots, p$, have been computed by the p parallel processors, the affine hull of the points $\{x^i, x^{i_1}, \ldots, x^{i_p}\}$ is typically searched for a best point by a synchronization step as follows.

Synchronization.

$$\begin{aligned} x^{i+1} &= \mu_0^i x^i + \textstyle\sum_{k=1}^p \mu_k^i x^{i_k} \\ (\mu_0^i, \mu_1^i, \ldots, \mu_p^i) &\in \arg\min_{\mu_0, \mu_1, \ldots, \mu_p} f\left(\mu_0 x^i + \sum_{k=1}^p \mu_k x^{i_k}\right) \\ &\text{such that} \quad \mu_0 x^i + \sum_{k=1}^p \mu_k x^{i_k} \in X \\ &\qquad\qquad \mu_0 + \sum_{k=1}^p \mu_k = 1. \end{aligned} \tag{1.4}$$

In fact, our convergence results only require the synchronization step to generate a point whose objective value is no worse than the best value generated by the processors. We note immediately that a fundamental difference between our method and that of block Jacobi for solving nonlinear equations [4] and coordinate descent [19] is the presence of the "forget-me-not" term $x_{\bar{l}}^i + D_{\bar{l}}^i \lambda_{\bar{l}}$ in problem (1.2). Thus in contrast with block Jacobi and coordinate descent where the coordinates of the blocks of variables $x_{\bar{l}}$ handled by the other processors are held fixed, our parallel subproblems (1.2) allow these blocks of variables to move in arbitrary directions $D_{\bar{l}}^i$, that can be generated, for example, using steepest-descent or quasi-Newton directions in the space of these variables. This novel idea allows each parallel processor to obtain a better minimum and endows the algorithm with a robustness property that makes it difficult to fail, as seen from our numerical tests.

We now outline the contents of the paper. In §2 we consider the unconstrained case and establish convergence of our algorithm by using ideas similar to those of the parallel gradient algorithm proposed in [13]. We also establish a linear convergence rate for the unconstrained algorithm. In §3 we consider problems with block separable constraints, that is, where the constraints are a Cartesian product of closed convex sets. In §4 we consider more general constraints, and in §5 we give some preliminary computational results. We conclude with brief remarks in §6.

We briefly describe our notation. For a differentiable function $f: \mathrm{R}^n \to \mathrm{R}$, ∇f denotes the n-dimensional (row) vector of partial derivatives with respect to x, and $\nabla_l f$ denotes the n_l-dimensional vector of partial derivatives with respect to $x_l \in \mathrm{R}^{n_l}$, $l = 1, \ldots, p$. If f has continuous first partial derivatives on R^n, we say $f \in C^1(\mathrm{R}^n)$. If f has Lipschitz continuous first partial derivatives on R^n with constant $K > 0$, that is,

$$\|\nabla f(y) - \nabla f(x)\| \leq K \|y - x\| \quad \forall x, y \in \mathrm{R}^n,$$

we write $f \in LC^1_K(\mathrm{R}^n)$. Unless stated otherwise, $\|\cdot\|$ denotes the Euclidean norm, that is, $\|x\| = \sqrt{x^T x}$ for x in a finite-dimensional space of appropriate dimension. A differentiable function $f: \mathrm{R}^n \to \mathrm{R}$ is strongly convex on R^n with constant $k > 0$ if

$$f(y) - f(x) - \nabla f(x)(y - x) \geq \frac{k}{2} \|y - x\|^2 \quad \forall x, y \in \mathrm{R}^n$$

or, equivalently,

$$(\nabla f(y) - \nabla f(x))(y - x) \geq k \|y - x\|^2 \quad \forall x, y \in \mathrm{R}^n.$$

We adapt the definition of a forcing function [16, p. 479] for our purposes.

DEFINITION 1.1. Forcing Function. *A forcing function σ is a continuous function from the nonnegative real line R_+ into itself such that $\sigma(0) = 0$, $\sigma(\xi) > 0$, for $\xi > 0$ and for every sequence of nonnegative real numbers $\{\xi_i\}$:*

$$\sigma(\xi_i) \to 0 \text{ implies } \xi_i \to 0.$$

Some typical examples of forcing functions are

$$\gamma\xi, \gamma\xi^2, \max\{\sigma_1(\xi), \sigma_2(\xi)\}, \min\{\sigma_1(\xi), \sigma_2(\xi)\} \quad \text{and} \quad \sigma_2(\sigma_1(\xi)),$$

where γ is a positive number and $\sigma_1(\xi)$ and $\sigma_2(\xi)$ are forcing functions.

2. Unconstrained parallel variable distribution. We begin by stating and establishing convergence of a parallel variable distribution (PVD) algorithm for the unconstrained minimization problem (1.1), where $X = \mathrm{R}^n$.

ALGORITHM 2.1. *Start with any $x^0 \in \mathrm{R}^n$. Having x^i, stop if $\nabla f(x^i) = 0$. Otherwise, compute x^{i+1} as follows.*

Parallelization: *For $l = 1, \ldots, p$ compute $x^{i_l} \in \mathrm{R}^n$ such that*

$$(y_l^i, \lambda_{\bar{l}}^i) \in \arg\min_{x_l, \lambda_{\bar{l}}} f(x_l, x_{\bar{l}}^i + D_{\bar{l}}^i \lambda_{\bar{l}}), \qquad x^{i_l} := (y_l^i, x_{\bar{l}}^i + D_{\bar{l}}^i \lambda_{\bar{l}}^i). \tag{2.1}$$

Here $D_{\bar{l}}^i$ is an $n_{\bar{l}}$-by-$(p-1)$ matrix generated from an arbitrary direction $d^i \in \mathrm{R}^n$ as in (1.3).

One particular choice is $d^i = -\nabla f(x^i) / \|\nabla f(x^i)\|$.

Synchronization: *Compute $x^{i+1} \in \mathrm{R}^n$ such that*

$$f(x^{i+1}) \leq \min_{l=1,\ldots,p} f(x^{i_l}). \tag{2.2}$$

For example, x^{i+1} can be computed as in (1.4).

We now establish convergence of the algorithm to a stationary point x, that is, $\nabla f(x) = 0$.

THEOREM 2.2. Convergence of the unconstrained PVD algorithm *Let* $f \in LC^1_K(\mathrm{R}^n)$. *For a bounded sequence* $\{d^i\}$, *either the sequence* $\{x^i\}$ *generated by Algorithm* 2.1 *terminates at a stationary point* $x^{\bar{i}}$ *or each of its accumulation points is stationary and* $\lim_{i\to\infty} \nabla f(x^i) = 0$.

Proof. For $l = 1, \ldots, p$ define the auxiliary functions $\theta^i_l : \mathrm{R}^{n_l+(p-1)} \to \mathrm{R}$, as follows:

$$\theta^i_l(x_l, \lambda_{\bar{l}}) := f(x_l, x^i_{\bar{l}} + D^i_{\bar{l}}\lambda_{\bar{l}}). \tag{2.3}$$

Then

$$\nabla\theta^i_l(x_l, \lambda_{\bar{l}}) = \left[\nabla_l f(x_l, x^i_{\bar{l}} + D^i_{\bar{l}}\lambda_{\bar{l}}), \nabla_{\bar{l}} f(x_l, x^i_{\bar{l}} + D^i_{\bar{l}}\lambda_{\bar{l}}) D^i_{\bar{l}}\right]. \tag{2.4}$$

Since f has a Lipschitz continuous gradient and $\{d^i\}$ is bounded, it follows that θ^i_l also has a Lipschitz continuous gradient with constant K_1, say. Define now, for $l = 1, \ldots, p$, $(z^i_l, \nu^i_l) \in \mathrm{R}^{n_l+(p-1)}$ as follows:

$$(z^i_l, \nu^i_l) := (x^i_l, 0) - \frac{1}{K_1}\nabla\theta^i_l(x^i_l, 0). \tag{2.5}$$

By quadratic bound Lemma A.2 of Appendix A, it follows that for $l = 1, \ldots, p$,

$$\theta^i_l(x^i_l, 0) - \theta^i_l(z^i_l, \nu^i_l) \geq \frac{1}{2K_1}\left\|\nabla\theta^i_l(x^i_l, 0)\right\|^2. \tag{2.6}$$

But by the definition of the PVD algorithm parallelization step (2.1),

$$\theta^i_l(y^i_l, \lambda^i_{\bar{l}}) \leq \theta^i_l(z^i_l, \nu^i_l). \tag{2.7}$$

Hence the last two inequalities yield for $l = 1, \ldots, p$

$$\theta^i_l(x^i_l, 0) - \theta^i_l(y^i_l, \lambda^i_{\bar{l}}) \geq \frac{1}{2K_1}\left\|\nabla\theta^i_l(x^i_l, 0)\right\|^2 \tag{2.8}$$

or

$$f(x^i) - f(y^i_l, x^i_{\bar{l}} + D^i_{\bar{l}}\lambda^i_{\bar{l}}) \geq \frac{1}{2K_1}\left\|\nabla\theta^i_l(x^i_l, 0)\right\|^2 \geq \frac{1}{2K_1}\left\|\nabla_l f(x^i)\right\|^2. \tag{2.9}$$

By noting the definition of x^{i_l} in (2.1), we have

$$f(x^i) - f(x^{i_l}) \geq \frac{1}{2K_1}\left\|\nabla_l f(x^i)\right\|^2, \qquad l = 1, \ldots, p. \tag{2.10}$$

Hence summing over l and dividing by p gives

$$f(x^i) - \frac{1}{p}\sum_{l=1}^{p} f(x^{i_l}) \geq \frac{1}{2pK_1}\sum_{l=1}^{p}\left\|\nabla_l f(x^i)\right\|^2 = \frac{1}{2pK_1}\left\|\nabla f(x^i)\right\|^2. \tag{2.11}$$

But from the synchronization step (2.2), we have the inequality

$$f(x^{i+1}) \leq \frac{1}{p}\sum_{l=1}^{p} f(x^{i_l}). \tag{2.12}$$

Combining the last two inequalities gives

$$f(x^i) - f(x^{i+1}) \geq \frac{1}{2pK_1} \left\|\nabla f(x^i)\right\|^2. \tag{2.13}$$

By definition, the PVD algorithm terminates at an $x^{\bar{i}}$ only if $\nabla f(x^{\bar{i}}) = 0$. Suppose now $\nabla f(x^{\bar{i}}) \neq 0$ for all i and that $\{x^{i_j}\}$ converges to $\bar{x}$. Since the sequence $\{f(x^i)\}$ is nonincreasing by (2.13) and has an accumulation point $f(\bar{x})$, it converges to $f(\bar{x})$ by Lemma A.1 of Appendix A. By (2.13), we have that

$$0 = \lim_{j\to\infty} (f(x^{i_j}) - f(x^{i_j+1})) \geq \lim_{j\to\infty} \frac{1}{2pK_1} \left\|\nabla f(x^{i_j})\right\|^2. \tag{2.14}$$

Hence $\lim_{j\to\infty} \left\|\nabla f(x^{i_j})\right\|^2$ and $\nabla f(\bar{x}) = 0$. The limiting property of the sequence $\{\nabla f(x^i)\}$ also follows from (2.13). □

If we further assume that the function f is strongly convex on R^n, then the sequence $\{x^i\}$ generated by the PVD algorithm converges linearly. We state the result as the following theorem.

THEOREM 2.3. Linear convergence of PVD algorithm. *If in addition to the assumptions of Theorem* 2.2, *the function f is strongly convex with constant $k > 0$, then the sequence of iterates $\{x^i\}$ generated by the* PVD *algorithm converges linearly to the unique solution $\bar{x}$ of* (1.1) *at the linear root rate*

$$\left\|x^i - \bar{x}\right\| \leq \left(\frac{2}{k}(f(x^0) - f(\bar{x}))\right)^{\frac{1}{2}} \left(1 - \frac{1}{p}\left(\frac{k}{K_1}\right)^2\right)^{\frac{i}{2}}. \tag{2.15}$$

Proof. The proof follows from (2.13) and the linear convergence theorem, Theorem A.3 of Appendix A. □

We turn our attention now to optimization problems with block separable constraint sets.

3. PVD with block separable constraints. We consider the following problem in this section:

$$\min_{x\in X} f(x) = \min_{\substack{x_l\in X_l \\ l=1,\ldots,p}} f(x_1, \ldots, x_p). \tag{3.1}$$

That is, $x \in \prod_{l=1}^p X_l$, where X_l, $l = 1, \ldots, p$ are closed convex sets in R^{n_l}, $l = 1, \ldots, p$, $\sum_{l=1}^p n_l = n$.

Before we specify our PVD algorithm for this problem above, we need a few definitions and some preliminary results. We begin with the concept of an optimality function.

DEFINITION 3.1. Optimality function. *For the problem* $\min_{x\in X} f(x)$, $f\colon \mathrm{R}^n \to \mathrm{R}$ *and* $X \subseteq \mathrm{R}^n$, *the function* $\omega\colon X \to \mathrm{R}$ *is an* optimality function *if*:

(i) $\omega(x) \geq 0$ *on* X;

(ii) $x \in \arg\min_{x\in X} f(x) \Longrightarrow \omega(x) = 0$;

(iii) ω *is nonconstant and lower semicontinuous on* X.

A stationary point *for the optimality function is an* $x \in X$ *such that* $\omega(x) = 0$.

The nonconstancy condition is merely to rule out the trivial case of $\omega(x) \equiv 0$. If $X = \mathrm{R}^n$ and $f \in C^1(\mathrm{R}^n)$, then $\omega(x) := \|\nabla f(x)\|$ is an optimality function for $\min_{x\in\mathrm{R}^n} f(x)$, and the notion of a stationary point is standard. If X is a closed

convex set in R^n, then the following function determined from the minimum principle [12], [7] serves as the *minimum principle* optimality function

$$\phi(x) := -\min_d \{\nabla f(x)d \mid x + d \in X, \|d\|_\infty \leq \alpha\} \quad \text{for some } \alpha > 0. \tag{3.2}$$

This is closely related to the gap function for convex programs [11]. A simple argument shows that ϕ is lower semicontinuous on X when $f \in C^1(\mathrm{R}^n)$. Furthermore, if $\bar{x}$ is a stationary point for ϕ, then

$$0 = \min_{y \in X} \{\nabla f(\bar{x})(y - \bar{x}) \mid \|y - \bar{x}\|_\infty \leq \alpha\}.$$

Hence, $\bar{x} \in \arg\min_{y \in X} \{\nabla f(\bar{x})(y - \bar{x}) \mid \|y - \bar{x}\|_\infty \leq \alpha\}$. Since the objective function is linear in y and X is convex, it follows that $\bar{x} \in \arg\min_{y \in X} \nabla f(\bar{x})(y - \bar{x})$ also, or equivalently that

$$\bar{x} \in X \quad \text{and} \quad \nabla f(\bar{x})(y - \bar{x}) \geq 0 \quad \forall y \in X. \tag{3.3}$$

This is precisely the minimum principle necessary optimality condition and we summarize the discussion above as the following lemma.

LEMMA 3.2. *If $\bar{x}$ is a stationary point for the optimality function ϕ given in* (3.2), *then $\bar{x}$ satisfies the minimum principle* (3.3).

We show now that using an Armijo stepsize rule [1] along a bounded Frank–Wolfe algorithm direction [9] produces a function decrease that dominates the minimum principle optimality function (3.2). This relationship will be needed in establishing the convergence of our PVD algorithm. However, we emphasize that the PVD algorithm does not employ either the Frank–Wolfe algorithm or the Armijo stepsize.

LEMMA 3.3. *For the problem $\min_{x \in X} f(x)$, where X is a convex subset of R^n and $f \in LC^1_K(\mathrm{R}^n)$, consider the following direction-stepsize procedure for a given $x \in X$.*

Direction (Frank–Wolfe):

$$d \in \arg\min \{\nabla f(x)d \mid d \in X - x, \quad \|d\|_\infty \leq \alpha\}. \tag{3.4}$$

Stepsize (Armijo): $\nu = \max\{1, \frac{1}{2}, \frac{1}{4}, \ldots\}$ *such that*

$$f(x) - f(x + \nu d) \geq -\frac{\nu}{2} \nabla f(x)d. \tag{3.5}$$

Then for

$$x^+ := x + \nu d \tag{3.6}$$

it follows that

$$f(x) - f(x^+) \geq \sigma(\phi(x)), \tag{3.7}$$

where ϕ is the minimum principle optimality function defined in (3.2), *σ is the forcing function given by*

$$\sigma(\xi) := \min\left\{\frac{1}{2}\xi, \frac{1}{4\alpha^2 K}\xi^2\right\}. \tag{3.8}$$

Proof. By the quadratic bound Lemma A.2 of Appendix A,

$$\begin{aligned} f(x) - f(x+\nu d) &\geq \nu\left(-\nabla f(x)d - \frac{K\nu}{2}\|d\|^2\right) \\ &= \nu\left(-\frac{1}{2}\nabla f(x)d - \frac{K\nu}{2}\|d\|^2 - \frac{1}{2}\nabla f(x)d\right) \\ &\geq -\frac{\nu}{2}\nabla f(x)d, \end{aligned}$$

whenever $\frac{K\nu}{2}\|d\|^2 \leq -\frac{1}{2}\nabla f(x)d$. Hence by the stepsize choice, either $\nu = 1$, or else when ν is replaced by 2ν the last inequality is violated, that is,

$$K\nu\|d\|^2 > -\frac{1}{2}\nabla f(x)d.$$

Since the direction generated by (3.4) satisfies $\|d\| \leq \alpha$, it follows from the last inequality and the possibility that $\nu = 1$, that

$$\nu \geq \min\left\{1, -\frac{1}{2\alpha^2 K}\nabla f(x)d\right\}. \tag{3.9}$$

Using this lower bound on ν in (3.5) gives the desired inequality (3.7) after replacing $-\nabla f(x)d$ by $\phi(x)$ using (3.2). □

Before stating and proving convergence of our PVD algorithm, we need to define a distributed optimality function over the block separable constraint set.

DEFINITION 3.4. Distributed optimality function. *For the problem* (3.1), *we define the* distributed optimality function

$$\omega(x) := \sum_{l=1}^{p}\omega_l(x), \tag{3.10}$$

where $\omega_l: X \to \mathrm{R}$ *and for* $l = 1, \ldots, p$:

(i) $\omega_l(x) \geq 0$ *on* X,

(ii) $x \in \arg\min_{y_l \in X_l} f(y_l, x_{\bar{l}}) \Longrightarrow \omega_l(x) = 0$,

(iii) ω_l *is nonconstant and lower semicontinuous on* X.

A stationary point *for the optimality function* (3.10) *is an* $x \in X$ *such that* $\omega(x) = 0$.

We note that

$$x \in \arg\min_{x\in X} f(x) \Longrightarrow \omega(x) = \sum_{l=1}^{p}\omega_l(x) = 0.$$

An example of a distributed optimality function for problem (3.1) is the following:

$$\phi_l(x) := -\min_{\substack{d_l \in X_l - x_l \\ \|d_l\|_\infty \leq \alpha}} \nabla_l f(x) d_l. \tag{3.11}$$

The following lemma shows that a forcing function for the distributed optimality function (3.10) also forces any constituent partial optimality function to zero.

LEMMA 3.5. *If* σ *is a forcing function, then for any* $J \subseteq \{1, \ldots, p\}$,

$$\sigma(\omega(x^i)) \to 0 \Longrightarrow \sum_{l\in J}\omega_l(x^i) \to 0,$$

where ω is the distributed optimality function given by (3.10).

Proof. Since σ is a forcing function, it follows that

$$\sigma(\omega(x^i)) \to 0 \Longrightarrow \omega(x^i) = \sum_{l=1}^{p} \omega_l(x^i) \to 0.$$

However, by definition, $\omega_l(x^i) \geq 0$, from which the result follows. □

We are ready to state our PVD algorithm for (3.1) and establish its convergence.

ALGORITHM 3.6. *Start with any $x^0 \in \mathrm{R}^n$. Having x^i, stop if $\phi(x^i) = 0$. Otherwise, compute x^{i+1} as follows.*

Parallelization: *For $l = 1, \ldots, p$ compute $x^{i_l} \in \mathrm{R}^n$ such that*

$$\begin{aligned} &(y_l^i, \lambda_{\bar{l}}^i) \in \arg\min_{x_l, \lambda_{\bar{l}}} f(x_l, x_{\bar{l}}^i + D_{\bar{l}}^i \lambda_{\bar{l}}) \\ &\qquad \text{such that } x_l \in X_l, \quad x_{\bar{l}}^i + D_{\bar{l}}^i \lambda_{\bar{l}} \in \prod_{\substack{s=1 \\ s \neq l}}^{p} X_s \\ &x^{i_l} := (y_l^i, x_{\bar{l}}^i + D_{\bar{l}}^i \lambda_{\bar{l}}^i) \end{aligned} \tag{3.12}$$

Synchronization: *Compute $x^{i+1} \in X$ such that*

$$f(x^{i+1}) \leq \min_{l=1,\ldots,p} f(x^{i_l}). \tag{3.13}$$

THEOREM 3.7. Convergence of the block constrained PVD algorithm. *Let $f \in LC_K^1(\mathrm{R}^n)$ and let X_l be nonempty convex sets in R^{n_l} for $l = 1, \ldots, p$. For a bounded sequence $\{d^i\}$, either the sequence $\{x^i\}$ generated by Algorithm* 3.6 *terminates at a stationary point $x^{\bar{i}}$ or every accumulation point of $\{x^i\}$ is stationary; that is, it satisfies the minimum principle necessary optimality condition* (3.3) *and* $\lim_{i \to \infty} \phi(x^i) = 0$.

Proof. First of all, note that by choosing $x_l = x_l^i$ and $\lambda_{\bar{l}} = 0$ in (3.12), it follows that $f(x^{i_l}) \leq f(x^i)$ for $l = 1, \ldots, p$. Furthermore, $f(x^{i+1}) \leq f(x^{i_l})$ for all l by (3.13). Hence the sequence $\{f(x^i)\}$ is nonincreasing. If $\{x^i\}$ does not terminate at a stationary point $x^{\bar{i}}$ of the minimum principle optimality function, then consider any subsequence $\{x^{i_j}\}$ that converges to $\bar{x}$. Since $\{f(x^i)\}$ has an accumulation point $f(\bar{x})$, it must converge to $f(\bar{x})$ by Lemma A.1 of Appendix A. In particular, it follows that $f(x^i) - f(x^{i+1}) \to 0$. To complete the proof of the theorem, we will show that this implies that $\phi(x^i) \to 0$. Since ϕ is lower semicontinuous, it follows that $\phi(\bar{x}) = 0$ and by Lemma 3.2 the minimum principle (3.3) is satisfied at $\bar{x}$.

As in the proof of Theorem 2.2, we define the auxiliary functions $\theta_l^i \colon \mathrm{R}^{n_l + (k-1)} \to \mathrm{R}$ for $l = 1, \ldots, p$ by

$$\theta_l^i(x_l, \lambda_{\bar{l}}) := f(x_l, x_{\bar{l}}^i + D_{\bar{l}}^i \lambda_{\bar{l}}),$$

with gradient determined by

$$\nabla \theta_l^i(x_l, \lambda_{\bar{l}}) = \left[\nabla_l f(x_l, x_{\bar{l}}^i + D_{\bar{l}}^i \lambda_{\bar{l}}), \nabla_{\bar{l}} f(x_l, x_{\bar{l}}^i + D_{\bar{l}}^i \lambda_{\bar{l}}) D_{\bar{l}}^i \right].$$

Using the boundedness of $\{d^i\}$ we conclude that these gradients are Lipschitzian with constant K_1, say. Define now, for $l = 1, \ldots, p$, (z_l^i, ν_l^i) by an Armijo step (3.5) along

the bounded Frank–Wolfe direction (3.4) applied to the parallel subproblem (3.12) at the point x^i:

$$\min_{x_l,\lambda_{\bar{l}}} \left\{ f(x_l, x_{\bar{l}}^i + D_{\bar{l}}^i \lambda_{\bar{l}}) \;\middle|\; x_l \in X_l, \;\; x_{\bar{l}}^i + D_{\bar{l}}^i \lambda_{\bar{l}} \in \prod_{\substack{s=1 \\ s \neq l}}^{p} X_s \right\}$$

$$= \min_{x_l,\lambda_{\bar{l}}} \left\{ \theta_l^i(x_l, \lambda_{\bar{l}}) \;\middle|\; x_l \in X_l, \;\; x_{\bar{l}}^i + D_{\bar{l}}^i \lambda_{\bar{l}} \in \prod_{\substack{s=1 \\ s \neq l}}^{p} X_s \right\}. \tag{3.14}$$

By Lemma 3.3, we have that for $l = 1, \ldots, p$

$$\theta_l^i(x_l^i, 0) - \theta_l^i(z_l^i, \nu_l^i) \geq \sigma_l(\psi^l(x^i)), \tag{3.15}$$

where $\psi^l(x^i)$ represents the minimum principle optimality function associated with (3.14) and σ_l is the forcing function defined by (3.8). But by the definition of the parallelization step (3.12) of the PVD algorithm,

$$\theta_l^i(y_l^i, \lambda_l^i) \leq \theta_l^i(z_l^i, \nu_l^i). \tag{3.16}$$

Hence the last two inequalities yield for $l = 1, \ldots, p$

$$\theta_l^i(x_l^i, 0) - \theta_l^i(y_l^i, \lambda_l^i) \geq \sigma_l(\psi^l(x^i)) \tag{3.17}$$

or

$$f(x^i) - f(y_l^i, x_{\bar{l}}^i + D_{\bar{l}}^i \lambda_{\bar{l}}^i) \geq \sigma_l(\psi^l(x^i)). \tag{3.18}$$

Noting the definition of x^{i_l} in (3.12), we have

$$f(x^i) - f(x^{i_l}) \geq \sigma_l(\psi^l(x^i)), \qquad l = 1, \ldots, p. \tag{3.19}$$

Thus, for $l = 1, \ldots, p$

$$f(x^i) - f(x^{i_l}) \to 0 \Longrightarrow \sigma_l(\psi^l(x^i)) \to 0.$$

By invoking Lemma 3.5, it follows from the particular form of $\psi^l(x^i)$ for (3.14) that for $l = 1, \ldots, p$

$$\sigma_l(\psi^l(x^i)) \to 0 \Longrightarrow \phi_l(x^i) \to 0,$$

where $\phi_l(x^i) = -\min_{y_l \in X_l} \nabla_l f(x^i)(y_l - x_l^i)$. But since $\sum_{l=1}^{p} \phi_l(x^i) = -\min_{y \in X} \nabla f(x^i)(y - x^i) = \phi(x^i)$, we have that

$$\max_{l=1,\ldots,p} \{ f(x^i) - f(x^{i_l}) \} \to 0 \Longrightarrow \phi(x^i) \to 0$$

or, equivalently,

$$f(x^i) - \min_{l=1,\ldots,p} f(x^{i_l}) \to 0 \Longrightarrow \phi(x^i) \to 0. \tag{3.20}$$

But from the synchronization step (3.13), we have

$$f(x^{i+1}) \leq \min_{l=1,\ldots,p} f(x^{i_l}). \tag{3.21}$$

Combining (3.20) and (3.21) gives

$$f(x^i) - f(x^{i+1}) \to 0 \Longrightarrow \phi(x^i) \to 0 \tag{3.22}$$

as required. □

We now turn our attention to a problem with a more general nonlinear constraint set.

4. PVD with general constraints. We consider in this section the problem

$$\min f(x) \text{ such that } g(x) \leq 0, \tag{4.1}$$

where $f: \mathrm{R}^n \to \mathrm{R}$ and $g: \mathrm{R}^n \to \mathrm{R}^m$ are $C^1(\mathrm{R}^n)$. It turns out that if the constraints are not separable in the sense of §3, having a stationary point that results from minimizing the objective function with respect to individual blocks of variables $x_1, \ldots, x_p$ and subject to the problem constraints $g(x) \leq 0$ does not result in a useful point as was the case for the unconstrained minimization problem of §2 or the separable constraint case of §3. This is easily illustrated by the following simple example in R^2:

$$\min x_1^2 + x_2^2 \text{ such that } x_1 + x_2 \geq 2. \tag{4.2}$$

This strongly convex problem has a unique global solution at $\bar{x}_1 = \bar{x}_2 = 1$. However, the point $x_1 = 0.5$, $x_2 = 1.5$ is a global minimum with respect to each of x_1 and x_2 separately, subject to the problem constraint. That is,

$$0.5 = \arg\min_{x_1} \left\{ x_1^2 + 2.25 \mid x_1 \geq 0.5 \right\},$$
$$1.5 = \arg\min_{x_2} \left\{ 0.25 + x_2^2 \mid x_2 \geq 1.5 \right\}.$$

This property is possessed by all points in the first quadrant lying on the constraint $x_1 + x_2 = 2$. None of these points are of any use except the solution $\bar{x}_1 = \bar{x}_2 = 1$. Therefore, it seems that the only sensible way to distribute variables for problems with inseparable constraints is to convert them to unconstrained problems or to problems with separable constraints. Obvious methods for doing so are exterior penalty [8] and augmented Lagrangian methods [17], [3] for handling inseparable constraints while leaving separable constraints as explicit ones. However, a disadvantage of exterior penalty is the unboundedness of the penalty parameter, while the augmented Lagrangian formulation essentially changes the minimization problem into a saddlepoint problem. An approach that avoids both of these difficulties is the dual differentiable exact penalty function [10] formulation

$$\max_{(x,u) \in \mathrm{R}^{n+m}, u \geq 0} \theta(x, u, \gamma) := L(x, u) - \frac{1}{2\gamma} \left\| \nabla_x L(x, u) \right\|^2, \tag{4.3}$$

where $L(x, u)$ is the standard Lagrangian for (4.1)

$$L(x, u) := f(x) + u^T g(x). \tag{4.4}$$

Various theorems [10] relate (4.3) to (4.1), but the key point is that the penalty parameter γ remains finite and the objective function is differentiable. Thus (4.3) appears to be a reasonable formulation of the problem (4.1) with inseparable constraints as one with separable constraints to which the algorithms of §3 are applicable.

Preliminary computational results for the PVD algorithm are given in the next section.

5. Computational results. In this section we report on some preliminary computational results with the PVD algorithm for unconstrained optimization. Our implementation is written in Split–C [6], a parallel extension of the C programming language primarily intended for distributed memory multiprocessors. Split–C is designed around two objectives. The first of these objectives is to capture certain useful elements of shared memory, message passing, and data parallel programming in a

familiar context, while eliminating the primary deficiences of each paradigm. The second objective is to provide efficient access to the underlying machine, which in our work is a Thinking Machines CM-5. In our implementation, shared memory is used to handle data associated with the current best solution and termination conditions. Split–C facilitates easy coding of the synchronization problem that obtains its data via message passing, while allowing the data for the subproblems to be physically distributed across the processors. Much of this can also be carried out using CMMD [18], the message passing library of the CM-5. However, Split–C enables the code to be written in a more readily portable manner.

The current implementation uses MINOS 5.4, a newer version of MINOS 5.0 [14], to solve both the parallel subproblems and the synchronization problem. MINOS uses a quasi-Newton approach for each of these problems. MINOS was chosen as the optimization tool since it is very reliable, efficient, and can be called easily as a subroutine. Furthermore, although other codes may be more efficient, for unconstrained optimization (for example, implementations that consider sparse Hessians or approximations), MINOS can also be used to solve large-scale constrained problems. This enables the unconstrained code used to report results in this section to be generalized for the algorithm given in §3.

The test problems used below are a subset of the problems from the constrained and unconstrained testing environment (CUTE) [5]. Both the problems and tools for linking algorithms with the problems are available via anonymous ftp. The problems are written in SIF (standard input format) and include many practical and large-scale examples.

We now give results in Tables 5.1–5.8. Tables 5.1–5.4 are on smaller-size problems for which our parallel code could run on one processor. The results reported are for two different choices of directions. In Tables 5.1 and 5.2 the results are obtained using a zero direction, which we refer to as PVD0. This is closely related to the parallel gradient algorithm [13]. Essentially, this corresponds to a form of the block Jacobi method with synchronization for solving the problems, but which is covered by our convergence analysis. Table 5.1 gives total time in seconds and the total number of function (and gradient) evaluations, while Table 5.2 reports the speedup efficiency as calculated by

$$\frac{\text{Time on one processor}}{(\text{Time on } p \text{ processors}) * p} * 100. \tag{5.1}$$

Tables 5.3 and 5.4 give the same results for an implementation using the auxiliary directions d^i generated using a diagonal scaling of the gradient. The diagonal scaling is determined using an extra evaluation of the gradient in each processor to determine an approximate size for the Hessian matrix corresponding to that index. The precise scaling used is the reciprocal of the ith component of the vector

$$\nabla f(x + e_i) - \nabla f(x),$$

where x is the current point and e_i represents a vector that has ones in the ith block of variables and zeros elsewhere. We label the corresponding tables with PVD. As above, Table 5.3 gives the times and function evaluations for PVD, while Table 5.4 reports the speedup efficiencies as calculated by (5.1) for PVD.

Tables 5.5–5.8 report on the same algorithms PVD0 and PVD, but with larger problem sizes. In fact we were unable to run these problem instances with only one or two blocks due to the lack of sufficient memory on the processors of the CM-5 for

TABLE 5.1
PVD0: *Time(s) and function evaluations.*

Problem (size)	1 proc		2 procs		4 procs		8 procs	
	sec	(fn evals)	sec	(fn evals)	sec	(fn evals)	sec	(fn evals)
ARWHEAD(500)	12	(50)	6	(60)	4	(81)	4	(105)
BRYBND(500)	32	(71)	17	(92)	10	(122)	7	(129)
DIXMAANB(300)	5	(52)	4	(65)	2	(87)	2	(132)
DQDRTIC(500)	19	(50)	14	(70)	5	(77)	4	(110)
DQRTIC(500)	270	(533)	120	(801)	69	(1149)	66	(1804)
FREUROTH(500)	1300	(1878)	880	(3627)	480	(4452)	320	(4870)
LIARWHD(500)	28	(76)	8	(77)	8	(121)	9	(200)
MOREBV(100)	63	(1399)	38	(2010)	29	(3199)	31	(6020)
NONDIA(500)	24	(71)	12	(105)	5	(90)	4	(131)
NONDQUAR(500)	29	(63)	16	(100)	9	(140)	5	(163)
PENALTY2(100)	93	(1897)	30	(1128)	19	(1257)	10	(903)
POWER(500)	45	(85)	21	(124)	9	(162)	7	(260)
QUARTC(500)	270	(533)	120	(801)	69	(1149)	66	(1804)
SINQUAD(500)	98	(149)	14	(82)	6	(75)	5	(94)
TOINTGSS(500)	15	(62)	9	(63)	4	(73)	4	(128)
TQUARTIC(500)	44	(95)	6	(57)	4	(60)	2	(68)
VAREIGVL(500)	26	(72)	16	(87)	9	(124)	8	(187)

TABLE 5.2
PVD0: *Speedup efficiencies.*

Problem (size)	2 procs	4 procs	8 procs
ARWHEAD(500)	96.8	68.2	37.5
BRYBND(500)	94.1	80.0	57.1
DIXMAANB(300)	72.9	60.7	30.4
DQDRTIC(500)	67.9	99.0	59.4
DQRTIC(500)	112.5	97.8	51.1
FREUROTH(500)	73.9	67.7	50.8
LIARWHD(500)	179.5	87.5	39.8
MOREBV(100)	82.9	54.3	25.4
NONDIA(500)	100.0	127.7	69.8
NONDQUAR(500)	90.6	84.3	67.1
PENALTY2(100)	155.0	122.4	116.2
POWER(500)	107.1	125.0	82.7
QUARTC(500)	112.5	97.8	51.1
SINQUAD(500)	350.0	437.5	235.6
TOINTGSS(500)	85.2	98.7	45.7
TQUARTIC(500)	338.5	297.3	220.0
VAREIGVL(500)	81.3	69.1	41.7

TABLE 5.3
PVD: *Time(s) and function evaluations.*

Problem (size)	1 proc		2 procs		4 procs		8 procs	
	sec	(fn evals)	sec	(fn evals)	sec	(fn evals)	sec	(fn evals)
ARWHEAD(500)	12	(50)	6	(62)	4	(77)	4	(97)
BRYBND(500)	33	(71)	25	(119)	11	(131)	11	(207)
DIXMAANB(300)	5	(52)	2	(60)	2	(81)	2	(104)
DQDRTIC(500)	19	(50)	6	(59)	4	(81)	4	(113)
DQRTIC(500)	280	(533)	110	(583)	52	(763)	46	(1198)
FREUROTH(500)	1400	(1878)	170	(462)	740	(6960)	970	(14250)
LIARWHD(500)	29	(76)	45	(141)	75	(680)	6	(145)
MOREBV(100)	65	(1399)	41	(2253)	0	()	0	()
NONDIA(500)	24	(71)	330	(3505)	7	(179)	0	()
NONDQUAR(500)	30	(63)	13	(103)	8	(130)	4	(132)
PENALTY2(100)	96	(1897)	42	(1590)	31	(1908)	20	(1702)
POWER(500)	46	(85)	17	(110)	8	(175)	7	(255)
QUARTC(500)	280	(533)	110	(583)	51	(763)	46	(1198)
SINQUAD(500)	100	(149)	15	(86)	56	(620)	9	(175)
TOINTGSS(500)	15	(62)	7	(60)	4	(73)	5	(120)
TQUARTIC(500)	46	(95)	7	(59)	5	(94)	4	(133)
VAREIGVL(500)	27	(72)	15	(89)	5	(89)	4	(106)

TABLE 5.4
PVD: *Speedup efficiencies.*

Problem (size)	2 procs	4 procs	8 procs
ARWHEAD(500)	96.8	68.2	37.5
BRYBND(500)	66.0	75.0	37.5
DIXMAANB(300)	113.0	68.4	34.2
DQDRTIC(500)	150.8	105.6	62.5
DQRTIC(500)	127.3	134.6	76.1
FREUROTH(500)	411.8	47.3	18.0
LIARWHD(500)	32.2	9.7	60.4
MOREBV(100)	79.3	52.4	38.7
NONDIA(500)	2.9	1.8	41.7
NONDQUAR(500)	115.4	92.6	83.3
PENALTY2(100)	114.3	77.4	60.0
POWER(500)	135.3	147.4	79.9
QUARTC(500)	127.3	137.3	76.1
SINQUAD(500)	333.3	44.6	142.0
TOINTGSS(500)	113.6	89.3	36.8
TQUARTIC(500)	328.6	250.0	155.4
VAREIGVL(500)	90.0	125.0	80.4

TABLE 5.5
PVD0: *Time(s) and function evaluations on larger problems.*

Problem (size)	4 procs		8 procs		16 procs		32 procs	
	sec	(fn evals)	sec	(fn evals)	sec	(fn evals)	sec	(fn evals)
ARWHEAD(1000)	14	(85)	11	(120)	9	(158)	10	(253)
BRYBND(1000)	28	(119)	26	(205)	20	(216)	27	(354)
DIXMAANB(1500)	31	(84)	25	(134)	23	(227)	29	(387)
DQDRTIC(1000)	28	(101)	10	(107)	10	(177)	8	(285)
FMINSURF(1024)	20	(102)	16	(139)	19	(239)	18	(283)
MOREBV(1000)	97	(250)	53	(366)	32	(597)	25	(1054)
NONDQUAR(1000)	31	(143)	24	(269)	17	(246)	26	(425)
POWER(1000)	35	(173)	20	(269)	31	(449)	92	(827)
SINQUAD(1000)	19	(83)	13	(93)	10	(116)	11	(163)
TOINTGSS(1000)	18	(90)	10	(115)	11	(222)	11	(379)
TQUARTIC(1000)	14	(68)	8	(75)	6	(99)	5	(146)
VAREIGVL(1000)	32	(135)	20	(184)	22	(249)	32	(394)
WOODS(1000)	20	(126)	14	(159)	15	(237)	41	(397)

TABLE 5.6
PVD0: *Speedup efficiencies on larger problems.*

Problem (size)	8 procs	16 procs	32 procs
ARWHEAD(1000)	63.6	38.9	17.9
BRYBND(1000)	53.8	35.0	13.0
DIXMAANB(1500)	62.0	33.7	13.4
DQDRTIC(1000)	141.4	71.4	41.2
FMINSURF(1024)	62.5	26.3	13.9
MOREBV(1000)	91.5	75.8	48.5
NONDQUAR(1000)	64.6	45.6	14.9
POWER(1000)	87.5	28.2	4.8
SINQUAD(1000)	73.1	47.5	21.6
TOINTGSS(1000)	92.8	40.9	20.5
TQUARTIC(1000)	87.5	63.6	35.0
VAREIGVL(1000)	80.0	36.4	12.5
WOODS(1000)	71.4	33.3	6.1

running MINOS on the subproblems. Thus Table 5.5 has results only for 4, 8, 16, and 32 processors. The calculations of speedup efficiency in Table 5.6 are carried out using the following form:

$$\frac{\text{Time on four processors} * 4}{(\text{Time on } p \text{ processors}) * p} * 100. \tag{5.2}$$

Finally, Table 5.7 gives the timings and function evaluations for the larger problems with the auxiliary directions generated by the scaling of the gradient direction. Table 5.8 gives the corresponding speedup efficiencies for PVD on the larger problems

TABLE 5.7
PVD: *Time(s) and function evaluations on larger problems.*

Problem (size)	4 procs		8 procs		16 procs		32 procs	
	sec	(fn evals)	sec	(fn evals)	sec	(fn evals)	sec	(fn evals)
ARWHEAD(1000)	15	(80)	8	(92)	9	(139)	9	(216)
BRYBND(1000)	32	(133)	26	(208)	26	(319)	33	(554)
DIXMAANB(1500)	27	(90)	16	(104)	13	(142)	17	(259)
DQDRTIC(1000)	14	(81)	9	(114)	10	(175)	11	(254)
FMINSURF(1024)	21	(100)	16	(135)	13	(172)	25	(300)
MOREBV(1000)	120	(390)	95	(835)	36	(919)	39	(1787)
NONDQUAR(1000)	42	(166)	12	(158)	12	(195)	11	(294)
POWER(1000)	41	(230)	24	(292)	24	(331)	80	(677)
SINQUAD(1000)	160	(623)	30	(247)	22	(271)	32	(449)
TOINTGSS(1000)	12	(76)	11	(122)	11	(181)	10	(282)
TQUARTIC(1000)	15	(89)	12	(146)	9	(209)	10	(351)
VAREIGVL(1000)	27	(118)	11	(116)	14	(181)	19	(342)
WOODS(1000)	41	(184)	12	(136)	16	(211)	29	(381)

TABLE 5.8
PVD: *Speedup efficiencies on larger problems.*

Problem (size)	8 procs	16 procs	32 procs
ARWHEAD(1000)	92.6	43.1	20.8
BRYBND(1000)	61.5	30.8	12.1
DIXMAANB(1500)	84.4	51.9	19.9
DQDRTIC(1000)	75.3	35.7	15.9
FMINSURF(1024)	65.6	40.4	10.5
MOREBV(1000)	63.2	83.3	38.5
NONDQUAR(1000)	175.0	87.5	47.7
POWER(1000)	85.4	42.7	6.4
SINQUAD(1000)	266.7	181.8	62.5
TOINTGSS(1000)	54.5	27.3	15.2
TQUARTIC(1000)	62.5	40.3	18.8
VAREIGVL(1000)	122.7	48.2	17.8
WOODS(1000)	170.8	64.1	17.7

using (5.2).

We note that there are some very high speedup efficiencies (over 437%) as well as some very low ones (4.8%). We believe that the reason for the low efficiency is that the problem size is not sufficiently large for the number of processors employed. More precisely, in order to attain high speedup efficiency, the ratio of the size of the synchronization subproblem to the size of the parallel subproblems (approximately p^2/n) must be small. This hypothesis is generally borne out by the numerical results.

Comparing PVD0 and PVD, it is difficult to say which is the clear winner. For example, comparing the speedup efficiencies for the larger problems (Tables 5.6 and

5.8), 62% of the efficiencies for PVD are higher that those for PVD0. Thus, there is a slight indication that PVD is more efficient.

We believe that further experimentation on much larger problems with and without constraints is warranted and might reveal a clearer indication of efficiency as well as better algorithmic strategies. These strategies include a variety of different choices for d^i in the parallel subproblems. In addition, asynchronous implementation might overcome some of the low efficiencies obtained by our synchronous algorithm.

Computational results for multicategory discrimination problems using an algorithm closely related to PVD0 can also be found in [2].

6. Conclusion. Blocks of variables of optimization problems were distributed among parallel processors with each processor taking primary responsibility for updating its assigned block, while not forgetting about the other variables by allowing them to vary in a restricted but plausible fashion. A synchronization scheme optimizes over the points obtained by the parallel processors. Explicit convergence results were given for the unconstrained case as well as for particular classes of constraints. Preliminary computational results indicate the proposed method has the potential for high parallelization speedup efficiency. An asynchronous implementation, whereby each parallel processor optimizes over the currently available points from the other processors, may further increase speedup efficiency. Further theoretical and computational studies are needed to obtain even faster parallel algorithms that take advantage of the powerful multiprocessors that are now available.

Appendix A. For convenience we state some of the results needed in the paper.

LEMMA A.1. *If the nonincreasing sequence of real numbers* $\{f_i\}$ *has an accumulation point* $\bar{f}$, *then* $\{f_i\}$ *converges to* $\bar{f}$.

Proof. We first prove that $\{f_i\}$ is bounded below. Let $f_{i_j} \to \bar{f}$. If $\{f_i\}$ is unbounded below then there exist $\bar{i}$, $\bar{j}$ for which we have

$$\begin{aligned} \bar{f} &> f_{\bar{i}} && \text{(since } \{f_i\} \text{ is unbounded below)}, \\ f_{\bar{i}} &\geq f_{i_{\bar{j}}} && \text{(since } \{f_i\} \text{ is nonincreasing)}, \\ f_{i_{\bar{j}}} &\geq \bar{f} && \text{(since } \{f_{i_j}\} \text{ converges to } \bar{f}), \end{aligned}$$

which is a contradiction. Hence $\{f_i\}$ is bounded below and it converges. It must converge to $\bar{f}$ because for a convergent sequence all accumulation points are identical to the limit of that sequence. □

LEMMA A.2. Quadratic Bound Lemma ([15, p. 144], [4, p. 639]). *Let* $f \in LC^1_K(\mathrm{R}^n)$, *then*

$$f(y)-f(x)-\nabla f(x)(y-x) \leq |f(y) - f(x) - \nabla f(x)(y-x)| \leq \frac{K}{2} \|y-x\|^2 \quad \forall x, y \in \mathrm{R}^n.$$

THEOREM A.3. Linear Convergence Theorem. *Let* $f: \mathrm{R}^n \to \mathrm{R}$, *let* $\{x^i\} \subseteq \mathrm{R}^n$, *let* $S = \{x \mid f(x) \leq f(x^0)\}$. *If* $f \in LC^1_K(S)$ *and*

$$f(x^i) - f(x^{i+1}) \geq \alpha \left\|\nabla f(x^i)\right\|^2 \tag{A.1}$$

for some $\alpha > 0$, $i = 0, 1, 2, \ldots$, *then every accumulation point* $\bar{x}$ *of* $\{x^i\}$ *is stationary, that is,* $\nabla f(\bar{x}) = 0$.

If, in addition, f is strongly convex with parameter $k > 0$, then $\{x^i\}$ converges to the unique solution $\bar{x}$ of $\min_{x \in \mathbf{R}^n} f(x)$ at the linear root rate

$$\|x^i - \bar{x}\| \leq \left(\frac{2}{k}(f(x^0) - f(\bar{x}))\right)^{\frac{1}{2}} \left(1 - \frac{2\alpha k^2}{K}\right)^{\frac{i}{2}}.$$

Proof. For the first part, let $x^{i_j} \to \bar{x}$. Then

$$f(x^{i_j}) - f(x^{i_j+1}) \geq f(x^{i_j}) - f(x^{i_j+1}) \geq \alpha \left\|\nabla f(x^{i_j})\right\|^2 \geq 0.$$

Taking the limit as $j \to \infty$ and invoking the continuity of ∇f, we get

$$0 = f(\bar{x}) - f(\bar{x}) \geq \alpha \|\nabla f(\bar{x})\|^2 \geq 0.$$

For the second part of the theorem, we note that strong convexity guarantees that S is compact and f is strictly convex. Hence $\{x^i\}$ must have at least one accumulation point $\bar{x}$ that minimizes $f(x)$ on $\mathbf{R}^n$. By the strict convexity of f and the first part of the theorem, every accumulation point must be the same and hence the sequence $\{x^i\}$ converges to $\bar{x}$, which is the unique solution of $\min_{x \in \mathbf{R}^n} f(x)$.

We establish now the linear root rate of convergence of $\{x^i\}$. We have

$$\begin{aligned}\|\nabla f(x^i)\| \|x^i - \bar{x}\| &= \|\nabla f(x^i) - \nabla f(\bar{x})\| \|x^i - \bar{x}\| \\ &\geq (\nabla f(x^i) - \nabla f(\bar{x}))(x^i - \bar{x}) \geq k \|x^i - \bar{x}\|^2,\end{aligned}$$

where the last inequality follows from the strong convexity of f. Hence (A.1) implies

$$f(x^i) - f(x^{i+1}) \geq \alpha k^2 \|x^i - \bar{x}\|^2.$$

Upon using quadratic bound Lemma A.2 of the appendix, the last inequality yields

$$f(x^i) - f(x^{i+1}) \geq \frac{2\alpha k^2}{K}(f(x^i) - f(\bar{x})).$$

This is equivalent to

$$\left(1 - \frac{2\alpha k^2}{K}\right)(f(x^i) - f(\bar{x})) \geq f(x^{i+1}) - f(\bar{x}).$$

By induction we have then

$$f(x^i) - f(\bar{x}) \leq \left(1 - \frac{2\alpha k^2}{K}\right)^i (f(x^0) - f(\bar{x})). \tag{A.2}$$

Again, from strong convexity we have that

$$f(x^i) - f(\bar{x}) \geq \frac{k}{2} \|x^i - \bar{x}\|^2 + \nabla f(\bar{x})(x^i - \bar{x})$$

or

$$\left(\frac{2}{k}(f(x^i) - f(\bar{x}))\right)^{\frac{1}{2}} \geq \|x^i - \bar{x}\|.$$

Using the last inequality in (A.2) gives

$$\|x^i - \bar{x}\| \leq \left(1 - \frac{2\alpha k^2}{K}\right)^{\frac{i}{2}} \left(\frac{2}{k}(f(x^0) - f(\bar{x}))\right)^{\frac{1}{2}}$$

as required. □

Acknowledgment. We are indebted to Ms. Chunhui Chen, a Ph.D. student of O. L. Mangasarian for valuable suggestions regarding the proofs of Theorems 2.2 and 3.7.

REFERENCES

[1] L. ARMIJO, *Minimization of functions having Lipschitz-continuous first partial derivatives*, Pacific J. Math., 16 (1966), pp. 1–3.

[2] K. P. BENNETT AND O. L. MANGASARIAN, *Serial and parallel multicategory separation*, SIAM J. Optim., 4 (1994), pp. 722–734.

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

[4] D. P. BERTSEKAS AND J. N. TSITSIKLIS, *Parallel and Distributed Computation*, Prentice-Hall, Englewood Cliffs, NJ, 1989.

[5] I. BONGARTZ, A. R. CONN, N. GOULD, AND PH. TOINT, *CUTE: Constrained and unconstrained testing environment*, Publications du Département de Mathématique Report 93/10, Facultés Universitaires de Namur, Namur, Belgium, 1993.

[6] D. CULLER, *The Split–C Programming Language*, Computer Science Department, University of California, Berkeley, 1993.

[7] M. C. FERRIS AND O. L. MANGASARIAN, *Minimum principle sufficiency*, Math. Prog., 57 (1992), pp. 1–14.

[8] A. V. FIACCO AND G. P. MCCORMICK, *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*, John Wiley & Sons, New York, 1968.

[9] M. FRANK AND P. WOLFE, *An algorithm for quadratic programming*, Naval Research Logistics Quart., 3 (1956), pp. 95–110.

[10] S.-P. HAN AND O. L. MANGASARIAN, *A dual differentiable exact penalty function*, Math. Prog., 25 (1983), pp. 293–301.

[11] D. W. HEARN, *The gap function of a convex program*, Oper. Res. Lett., 1 (1982), pp. 67–71.

[12] O. L. MANGASARIAN, *Nonlinear Programming*, McGraw-Hill, New York, 1969.

[13] ———, *Parallel gradient distribution in unconstrained optimization*, Tech. Report 1145, Computer Sciences Department, University of Wisconsin, Madison, Apr. 1993, SIAM J. Control Optim., to appear.

[14] B. A. MURTAGH AND M. A. SAUNDERS, *MINOS 5.0 user's guide*, Tech. Report SOL 83.20, Stanford University, Stanford, CA, 1983; MINOS 5.4 Release Notes, December 1992.

[15] J. M. ORTEGA, *Numerical Analysis, a Second Course*, Academic Press, New York, 1972.

[16] J. M. ORTEGA AND W. C. RHEINBOLDT, *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, New York, 1970.

[17] R. T. ROCKAFELLAR, *Augmented Lagrange multiplier functions and duality in nonconvex programming*, SIAM J. Control Optim., 12 (1974), pp. 268–285.

[18] THINKING MACHINES CORPORATION, CMMD *Reference Manual, Version* 3.0, Cambridge, MA, 1993.

[19] P. TSENG, *Dual coordinate ascent methods for non-strictly convex minimization*, Math. Prog., 59 (1993), pp. 231–248.

SIAM J. OPTIMIZATION
Vol. 4, No. 4, pp. 833–846, November 1994
© 1994 Society for Industrial and Applied Mathematics
009

PARALLEL FACTORIZATION OF STRUCTURED MATRICES ARISING IN STOCHASTIC PROGRAMMING*

ELIZABETH R. JESSUP†, DAFENG YANG‡, AND STAVROS A. ZENIOS†

Abstract. Solving the deterministic equivalent formulation of two-stage stochastic programs using interior point algorithms requires the solution of linear systems of the form

$$(AD^2A^\top)dy = b.$$

The constraint matrix A has a dual, block-angular structure. We develop a parallel matrix factorization procedure using the Sherman–Morrison–Woodbury formula, based on the work of Birge and Qi [*Management Sci.*, 34 (1988), pp. 1472–1479]. This procedure requires the solution of smaller, independent systems of equations. With the use of optimal communication algorithms and careful attention to data layout a parallel implementation is obtained that achieves near perfect speedup. Scalable performance is observed on an Intel iPSC/860 hypercube and a Connection Machine CM–5. Results are reported with the solution of the linear systems arising when solving stochastic programs with up to 98,304 scenarios, which correspond to deterministic equivalent linear programs with up to 1,966,090 constraints and 13,762,630 variables.

Key words. stochastic programming, large-scale optimization, interior point algorithms, parallel algorithms

AMS subject classifications. 65Y05, 68Q22, 90C15, 90C06

1. Introduction.

Stochastic programming is used to model a wide range of practical applications with uncertain input data. When the input data are discretely distributed — represented, for example, by a set of scenarios — the stochastic program can be formulated as a deterministic equivalent linear program with a dual, block-angular constraint matrix. For a large number of scenarios these linear programs could be extremely large and computationally intractable.

The literature on stochastic programming is extensive, dating back to the 1950s with the works of Dantzig and Beale and we do not stop to review it here. Significant progress has been made over the last decade. Developments in interior point algorithms hold great promise for the solution of large-scale programs of this form. Substantial improvements are also expected with the use of parallel computers. For an introduction to stochastic programming see Wets [15], and for applications see, for example, Birge and Holmes [3], Zenios [16], and their references.

Recent algorithmic approaches for solving stochastic programs are found in Rockafellar and Wets [13], Birge and Holmes [3], Mulvey and Ruszczyński [10], and Nielsen and Zenios [12]. Parallel algorithms are also discussed in [3], [10], and [12]. One strand of research [10], [12], [13] develops decomposition algorithms that solve the stochastic program by solving a sequence of smaller subproblems—one for each scenario—and

* Received by the editors October 8, 1993; accepted for publication (in revised form) April 21, 1994.

† Department of Computer Science, University of Colorado, Boulder, Colorado 80309-0430 (jessup@cs.colorado.edu). This author was funded by Department of Energy contract DE-FG02-92ER25122 and by National Science Foundation National Young Investigator Award.

‡ Operations and Information Management Department, The Wharton School, University of Pennsylvania, Philadelphia, Pennsylvania 19104 and Department of Public and Business Administration, University of Cyprus, Nicosia, Cyprus (yang32@wharton.upenn.edu and zenios@wharton.upenn.edu). These authors were funded by National Science Foundation grant CCR-91-04042. Partial support was also provided by the Universities of Urbino and Bergamo, Italy by a fellowship from the Gruppo Nazionales per L'Analisi Funzionala ed Applicazione and GNIM groups of Consiglio Nazionale delle Richerche (CNR).

combining the results through a master program or an aggregation step. A second strand of research [2], [3], [9] that is also pursued in this paper attacks the linear programming formulation of a stochastic program using interior point algorithms and seeks matrix factorization techniques that exploit the structure of the constraint matrix. Efficient techniques are sought for the computation of the dual step dy by solving the symmetric, positive definite system $(AD^2A^\top)dy = b$, where the constraint matrix A has a dual, block-angular structure.

Birge and Holmes [3] compare different methods for the solution of this system. They find that the factorization technique of Birge and Qi (BQ) [2] based on the Sherman–Morrison–Woodbury updating formula is more efficient, stable, and accurate than a method based on *problem reformulations* suggested by Lustig, Mulvey, and Carpenter [9], or on methods using Schur complements. This conclusion is supported by numerical experiments: BQ is up to 20 times faster than methods based on problem reformulation.

Birge and Holmes also suggest that BQ is suitable for parallel computation. However, their experiments—on a distributed network of DECstations—show only modest speedups. The best speedup is approximately 3.0 on four workstations. Adding more workstations worsens performance. This mediocre performance is caused by the distributed, networked environment they used by Birge and Holmes: communication is too expensive for the small-sized problems they could solve. In this paper we show that very good speedups are attainable using a "true" parallel machine (like the Intel iPSC/860 hypercube or the Connection Machine CM–5) and employing suitable communication algorithms. Almost linear speedups are achieved for a wide range of problem sizes and machine dimensions. In the process, we solve test problems with up to 98,304 scenarios, which correspond to linear programs with 1,966,090 constraints and 13,762,630 variables.

Factorization procedures similar to BQ were proposed by Choi and Goldfarb [4] for multicommodity network flow problems. Their worst case analysis shows substantial savings in computations, but the communication costs were ignored and no empirical results were reported. The techniques developed here are applicable to multicommodity network flow problems as well.

Section 2 establishes notation. Section 3 reviews the BQ matrix factorization procedure and identifies the basic parallelism and its communication requirements. Section 4 develops appropriate communication schemes on a hypercube and discusses the parallel implementation. Section 5 presents computational results on the Intel hypercube. Section 6 discusses the parallel implementation of the code on the Connection Machine CM–5 and reports results. Section 7 addresses the solution of very large problems from two applications, and Section 8 concludes the paper.

2. The problem formulation. The two-stage stochastic program determines an optimal first-stage decision vector $x_0 \in \Re^{n_0}$ with cost vector $c_0 \in \Re^{n_0}$ before some random coefficients are observed, and then it takes an optimal *recourse* action after the random coefficients become known. We assume $l = 1, 2, \ldots, N$ scenarios of the random coefficients: $c_l \in \Re^{n_l}$, $T_l \in \Re^{m_l \times n_0}$, $W_l \in \Re^{m_l \times n_l}$, $b_l \in \Re^{m_l}$. (For simplicity we assume that the probability of scenario l is incorporated in the cost coefficients c_l.) We also use $(A)_{i\cdot}$ and $(A)_{\cdot i}$ to denote the ith row and column of matrix A, respectively. With this notation the program is written as

$$\begin{aligned} &\text{Minimize } c_0^\top x_0 + \textstyle\sum_{l=1}^{N} c_l^\top y_l \\ &\text{Subject to:} \\ &\qquad A_0 x_0 \qquad\qquad = b_0 \end{aligned}$$

$$\begin{aligned} T_l\, x_0 + W_l y_l &= b_l \text{ for } l = 1, 2, \ldots, N, \\ x_0, \quad y_l &\geq 0. \end{aligned}$$

This problem has $n = n_0 + \sum_{l=1}^{N} n_l$ variables and $m = m_0 + \sum_{l=1}^{N} m_l$ constraints. A_0 and W_l are assumed to have full row rank, with $m_l \leq n_l$ for all $l = 0, 1, 2, \ldots, N$, and $n_0 \leq \sum_{l=1}^{N} n_l$.

The application of a primal-dual, path-following interior point algorithm to this linear program requires the repeated solution of symmetric, positive definite linear systems of the form

$$(AD^2A^\top)dy = b. \tag{1}$$

A is the constraint matrix of the two-stage stochastic program, i.e.,

$$A = \begin{pmatrix} A_0 & & & \\ T_1 & W_1 & & \\ \vdots & & \ddots & \\ T_N & & & W_N \end{pmatrix}.$$

The matrix $D^2 \in \Re^{n \times n}$ is positive definite and diagonal.

This system of equations calculates the dual step dy. Its solution accounts for more than 90% of the computation in practical applications of the algorithm. The derivation of this system can be found in most references on interior point algorithms, see, e.g., Birge and Holmes [3].

3. The BQ matrix factorization procedure. The procedure for solving (1) is based on the following result.

THEOREM 1. *Let* $M = AD^2A^\top$, $S = \operatorname{diag}\{S_0, S_1, \ldots, S_N\}$ *where* $S_l = W_l D_l^2 W_l^\top \in R^{m_l \times m_l}, l = 1, \ldots, N$, $S_0 = I$ *is an* $m_0 \times m_0$ *identity matrix and* $D_l \in R^{n_l \times n_l}$ *is the (diagonal) submatrix of* D *corresponding to the* l*th block. Also, let*

$$G_1 = (D_0)^{-2} + A_0^\top A_0 + \sum_{l=1}^{N} T_l^\top S_l^{-1} T_l, \tag{2}$$

$$G = \begin{bmatrix} G_1 & A_0^\top \\ -A_0 & 0 \end{bmatrix}, \quad U = \begin{pmatrix} A_0 & I \\ T_1 & 0 \\ \vdots & \vdots \\ T_N & 0 \end{pmatrix}, \quad V = \begin{pmatrix} A_0 & -I \\ T_1 & 0 \\ \vdots & \vdots \\ T_N & 0 \end{pmatrix}.$$

If A_0 *and* W_l, $l = 1, \ldots, N$, *have full row rank, then* M *and* $G_2 \equiv -A_0 G_1^{-1} A_0^\top$ *are invertible and*

$$M^{-1} = S^{-1} - S^{-1} U G^{-1} V^\top S^{-1}. \tag{3}$$

Proof. For the proof, see Birge and Holmes [3, pp. 257–258]. □

It is easy to verify using (3) that the solution of $Mdy = b$ is given by $dy = p - r$, where p is given by $Sp = b$, and r is found by solving

$$Gq = V^\top p \quad \text{and} \quad Sr = Uq. \tag{4}$$

The vector p can be computed componentwise by solving $S_l p_l = b_l$, for $l = 1, \ldots, N$. To solve for q we exploit the block structure of G:

$$Gq = \begin{bmatrix} G_1 & A_0^\top \\ -A_0 & 0 \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} = \begin{bmatrix} \hat{p}_1 \\ \hat{p}_2 \end{bmatrix}, \quad \text{where} \quad \begin{bmatrix} \hat{p}_1 \\ \hat{p}_2 \end{bmatrix} \equiv V^\top p. \tag{5}$$

Hence, we get

$$q_2 = -G_2^{-1}(\hat{p}_2 + A_0 G_1^{-1} \hat{p}_1), \tag{6}$$

$$q_1 = G_1^{-1}(\hat{p}_1 - A_0^\top q_2). \tag{7}$$

Once q is known, r can be computed componentwise.

In [3], the procedure defined by Theorem 1 is called `Finddy`. This procedure is well suited for parallel implementation because it relies largely on matrix subblock computations that can be performed independently of one another. We consider here only implementations for distributed-memory parallel computers, although the matrix factorization procedure is equally applicable to shared-memory machines. The computation begins with submatrices T_{l+1}, W_{l+1} and D_{l+1} and vector segment b_{l+1} located on node l (a *node* is a single processor together with its associated memory). Node l can compute S_{l+1} and proceed independently with all computations involving only these data. We use $l = 1, 2, \ldots, N$ to index scenarios for consistency with the stochastic programming literature and $l = 0, 1, \ldots, N-1$ to index nodes for consistency with the literature on hypercubes.

Interprocessor data communication is necessary at only three points in the parallel `Finddy` algorithm. After forming the terms $T_l^\top S_l^{-1} T_l$ from (2) in parallel, the nodes communicate to form the matrix G_1 from (2) and the vectors $\hat{p}_1$ and $\hat{p}_2$ from (5). All computations involving dense matrices G_1 and G_2 can be done serially on node 0.

Node 0 must then broadcast the computed vector q to all other nodes. The remaining steps require only the distributed data S_{l+1}, T_{l+1}, and p_{l+1} on node l and may be carried out with full parallelism. A final communication step accumulates the partial vectors $(dy)_{l+1}$ in node 0. This vector can then be made available to all nodes for use in the subsequent steps of an interior point algorithm that uses this procedure.

The parallel `Finddy` algorithm can be summarized as follows.

PROCEDURE 3.1 (Parallel `Finddy`$(S, A_0, T_1, \ldots, T_N, b, dy)$). *Begin with the following data distribution: Node l holds S_{l+1}, T_{l+1}, and b_{l+1}, $l = 0, \ldots, N-1$. Node 0 also holds A_0, S_0, D_0, and b_0.*

1. *(In parallel, solve $Sp = b$). On node 0 solve $S_0 p_0 = b_0$. In parallel (on nodes $l = 0, \ldots, N-1$), solve $S_{l+1} p_{l+1} = b_{l+1}$ for p_{l+1}.*

2. *(Solve $Gq = V^\top p$).*

(a) *In parallel on nodes $l = 0, \ldots, N-1$, solve $S_{l+1}(u_{l+1})_i = (T_{l+1})_{\cdot i}$ for $(u_{l+1})_i$, $i = 1, \ldots, n_0$.*

(b) *In parallel on nodes $l = 0, \ldots, N-1$, multiply $T_{l+1}^\top u_{l+1}$.*
Serially (on node 0), compute $\hat{p}_2$.
Communicate to form G_1 and $\hat{p}_1$ on node 0.

(c) *Serially (on node 0), solve $G_1 u = \hat{p}_1$ for u and set $v = \hat{p}_2 + A_0 u$.*

(d) *Serially (on node 0), form G_2 by solving $(G_1) w_i = (A_0^\top)_{\cdot i}$ for w_i, for $i = 1, \ldots, m_0$ and setting $G_2 = -A_0[w_1, \ldots, w_{m_0}]$.*

(e) *Serially (on node 0), solve $G_2 q_2 = -v$ for q_2, and solve $G_1 q_1 = \hat{p}_1 - A_0^\top q_2$ for q_1. Communicate to distribute q_1 to all nodes.*

3. *(In parallel, solve $Sr = Uq$). On node 0, set $r_0 = A_0 q_1 + q_2$. On nodes $l = 0, \ldots, N-1$, solve $S_{l+1} r_{l+1} = T_{l+1} q_1$ for r_{l+1}.*

4. (*In parallel, form dy.*) *On node* 0, *set* $dy_0 = p_0 - r_0$. *On nodes* $l = 0, \ldots, N-1$, *set* $(dy)_{l+1} = p_{l+1} - r_{l+1}$. *Communicate to gather the vector* $dy = (dy_0, \ldots, dy_N)$ *on node* 0.

The procedure has a "coordination" phase at steps 2(b)–2(e) that is executed serially. These steps could also be executed in parallel, but redundantly, on all nodes, so that each node calculates locally its own copy of q_1. Details of this organization are given in Procedure 6.1. Yet another alternative is to use parallel dense matrix techniques to execute steps 2(b)–2(e). This alternative has not been explored. It may be more appropriate for shared-memory parallel machines.

4. The parallel implementation for the hypercube multiprocessor. In this section, we discuss a parallel implementation designed explicitly for a hypercube multiprocessor. A hypercube is a distributed-memory multiple-instruction multiple-data (MIMD) message-passing parallel computer in which processors are connected according to a hypercube graph. A hypercube graph of dimension d is called a *d-cube* and has $p = 2^d$ nodes. The most efficient codes on hypercubes are those in which messages do not cross on communication wires. Therefore, we have used contention-free routing schemes that use only nearest neighbor communications.

The communication steps for the parallel implementation of `Finddy` require either broadcasting of data from one node to all others (*one-to-all* communication) or gathering of data distributed among processors. The data gathers can be either *all-to-all* or *all-to-one* depending on the intended use of the data and the speed of the gather routines. Thus, all of the communication steps in Procedure 3.1 can be based on two types of communication algorithms. The first type, termed a *spanning tree broadcast* (STB) [14], broadcasts data from node 0 to all others via the links of a spanning tree of the hypercube graph rooted at node 0. All communication takes place between nearest neighbors. Conversely, data distributed among nodes can be gathered into a single node by traversing the spanning tree from its leaves to its roots in a *spanning tree gather* (STG). The second communication routine is based on an *alternate direction exchange* (ADE) as described in [14]. For each of d communication steps in an ADE, the d-cube splits into a different pair of $(d-1)$-cubes, and the 2^{d-1} pairs of corresponding nodes from the two cubes exchange and accumulate their data sets. An ADE can be used to accumulate in all $p = 2^d$ nodes pk bytes of data initially distributed with k bytes per node. STB, STG, and ADE are optimal broadcast and gather routines in the sense that they take the minimum number of communication steps — d steps on a d-cube [14]. Minimizing the number of communication steps is important on present-day distributed-memory computers as the cost of communication is typically high in comparison to the cost of computation.

The basic structure of our implementation of the parallel `Finddy` algorithm follows. More details and a complexity analysis are presented in [6]. The initial data distribution is as assumed for Procedure 3.1. We assume that the number of scenarios N equals the number of processors p. If N and p differ, the scenarios and their accompanying tasks are distributed equally among the processors. If p does not divide N, remaining scenarios and tasks are distributed one per processor until all have been assigned.

Steps 1, 2(a), *and* 2(b). All nodes solve their systems for p_{l+1} and $(u_{l+1})_i$, $i = 1, \ldots, n_0$ in parallel. (Note that the matrices S_{l+1} need be factored only once to solve the $n_0 + 1$ systems.) All nodes then compute their matrices $\hat{T}_{l+1} = (T_{l+1}^{\top}(u_{l+1})_1, \ldots, T_{l+1}^{\top}(u_{l+1})_{n_0})$ in parallel.

The nodes then accumulate the matrix G_1 on node 0 by means of a *spanning*

tree global sum. This communication scheme is based on an STG but after a node receives the partial sums from all of its children, it adds them to its own partial sum, then sends the result to its parent. After the root node 0 receives the partial sums of its children, it sums them with $(D_0)^{-2} + A_0^\top A_0$ to form the matrix G_1. The global sum $\hat{p}_1 = A_0^\top p_0 + \sum_{l=1}^{N} \hat{t}_l$ is produced in node 0 at the same time. Our routine for efficiently computing G_1 and $\hat{p}_1$ is called GpGATHER.

Node 0 computes $\hat{p}_2 = -p_0$ serially.

Steps 2(c), 2(d), *and* 2(e). In our hypercube implementation, all computations involving the small order dense matrices G_1 and G_2 are done serially in node 0. This is more efficient than applying parallel dense matrix techniques for problems with a small number of first-stage variables compared to the number of second-stage variables and for problems when a large number of scenarios is solved at each node (i.e., when $N/p >> 1$). The problems used for all of our experiments have these characteristics. If the first-stage matrices are large we may consider alternative implementations using parallel dense linear algebra techniques to execute steps 2(c), 2(d), and 2(e).

Node 0 broadcasts the computed vector q_1 to all others via an STB.

Steps 3 *and* 4. Steps 3 and 4 are essentially completely parallel. Node 0 determines r_0, and all nodes find their matrix vector products $T_{l+1}q_{l+1}$ and solve their sparse systems.

In parallel, the nodes then compute the vector segments $(dy)_{l+1}$. As a final step in the algorithm, the nodes accumulate the full correction vector dy in node 0. This final step contributes little to the total time and can be excluded if the next iteration calls for a distributed vector.

In steps 1 and 2(a) our Fortran code uses the memory-efficient sparse system solver routines SUPFCT and SUPSLV of Ng and Peyton [11]. It uses the LAPACK routines DPOTRF and DPOTRS to factor and solve the dense linear systems involving matrix G_1 and the LAPACK routines DGETRF and DGETRS to factor and solve the dense linear systems involving G_2 [1]. Assembler BLAS routines are used wherever appropriate [5].

5. Computational results on the iPSC/860. We now summarize computational experiences with the implementation on the Intel iPSC/860 using random test matrices. One set of experiments illustrates the efficiency of the algorithm as the number of scenarios increases on a wide range of machine sizes. A second set illustrates how the performance of the code is influenced by the aspect ratio of the scenario matrices.

5.1. The random test problems. Random test matrices with a varying number of scenarios were generated with the problem generator GENSLP of Kall and Keller [7]. The nonzero entries of the matrices A_0 and W_l are uniformly distributed in the range [–112.0,128.0]. Two percent of the entries of these matrices are nonzero. The matrices T_l are dense. The problems are such that $m_1 = m_2 = \cdots = m_N$ and $n_1 = n_2 = \cdots = n_N$. (Such problems are quite standard in the stochastic programming literature.) The nonzero, diagonal elements of the matrix D^2 are uniformly distributed in [0,10]. We solved five test problems for each experiment and reported the average solution times. The variation in solution time over each set of five problems was never greater than 1%.

All times reported in this paper (for both the iPSC/860 and the CM–5) are for the factorization and solution of a single system of equations $(AD^2A^\top)dy = b$. Reported times exclude the time needed for data input or output but include the time for all

TABLE 1
Computation times (msec) *for problems of varying number of scenarios* N *with* $n_0 = m_0 = m_1 = 100$ *on the* iPSC/860.

N	p	step 1,2ab	step 2cde	step 3	step 4	T_{serial}/T_p	$T_{N/p}/T_p$
16	2	4479	230	48	2	4.8%	94.4%
	4	2239	230	24	1	9.0%	88.2%
	8	1120	230	12	0	15.6%	76.9%
	16	561	230	6	0	24.3%	59.8%
64	2	17904	230	192	4	1.2%	98.5%
	4	8952	230	96	2	2.4%	96.5%
	8	4476	230	48	1	4.7%	92.5%
	16	2238	230	24	0	8.6%	84.8%
128	2	35808	230	384	8	0.6%	99.2%
	4	17904	230	192	4	1.2%	98.1%
	8	8952	230	96	2	2.4%	95.8%
	16	4476	230	48	1	4.6%	91.1%
256	2	71616	230	768	16	0.3%	99.5%
	4	35808	230	384	8	0.6%	98.9%
	8	17904	230	192	4	1.2%	97.5%
	16	8952	230	96	2	2.4%	94.6%

other computations and communications. The solutions were all computed in double precision, and the same accuracy was achieved for the parallel implementations as for the serial implementation whenever a comparison was possible. Residual errors ranged from about 10^{-13} to 10^{-11} for most large test problems. For some extremely ill-conditioned matrices (with the diagonal entries of D^2 ranging from 10^{-28} to 10^{+28} in value), the residual error rose to 10^{-7} in the worst case.

5.2. Parallel efficiency with increasing number of scenarios N. For the first group of experiments we fix the parameters $m_0 = n_0 = m_1 = 100$, $n_1 = 1000$, and vary the number of scenarios N. The total time to run the parallel algorithm on p nodes can be expressed by $T_p = T_{N/p} + T_{\text{serial}} + T_{\text{comm}}$. The total time to run the algorithm on one node is $T_1 = pT_{N/p} + T_{\text{serial}}$. That is, $pT_{N/P}$ is the time for the fully parallelizable part of the one processor algorithm, and T_{serial} is the time for the part that cannot be implemented in parallel. T_{comm} is the total time spent in communication. *Speedup* can be written as $S_p = T_1/T_p$ and *efficiency* as $E_p = S_p/p = T_1/(pT_p)$.

The timing data presented in Table 1 show that T_{serial}, which roughly equals the time for steps 2(c), 2(d), and 2(e), is very small compared to $T_{N/p}$ for the parameters used in our experiments. Thus, the attainable speedup is most influenced by the percentage of time spent in communication.

Table 2 shows the relation of efficiency to the percentage of total time spent in communication (i.e., T_{comm}/T_p). A 4% or less communication time corresponds to a 90% or better efficiency. The percentage of time spent in communication grows as more processors are applied to a matrix of fixed size.

As is evident in Table 2, both $T_{N/p}$ and T_{comm} are essentially dependent on N. However, the system solutions that dominate the time for steps 1 and 2(a) require $O(m_1^3)$ operations compared to the $O(Nm_1)$ byte transfers that dominate T_{comm} [6]. Clearly the algorithm is most efficient when the number of scenarios per processor, N/p, is large, so that the computation time outweighs the increase in communication time.

TABLE 2
Communication times (msec) *and efficiencies for problems of varying number of scenarios* N, *with* $n_0 = m_0 = m_1 = 100$ *on the* iPSC/860.

N	p	T_{comm}	T_{comm}/T_p	E_p
16	2	39	0.8%	0.97
	4	74	2.9%	0.90
	8	110	7.5%	0.80
	16	151	16.0%	0.61
64	2	54	0.3%	0.99
	4	98	1.0%	0.97
	8	136	2.8%	0.93
	16	176	6.6%	0.85
128	2	74	0.2%	1.00
	4	128	0.7%	0.98
	8	171	1.8%	0.96
	16	214	4.3%	0.91
256	2	114	0.2%	1.00
	4	189	0.5%	0.99
	8	240	1.3%	0.98
	16	288	3.0%	0.95

TABLE 3
Communication times for problems of varying m_1 *with* $n0 = m0 = 100$ *and* $N = 128$ (msecs).

m_1	p	T_{comm}	T_{comm}/T_p	E_p
10	2	38	2.8%	0.88
	4	73	8.8%	0.70
	8	108	17.8%	0.49
	16	146	28.6%	0.29
20	2	44	2.0%	0.93
	4	83	6.5%	0.80
	8	119	14.3%	0.61
	16	158	25.1%	0.41
50	2	54	1.1%	0.97
	4	98	3.7%	0.90
	8	138	8.9%	0.78
	16	176	17.8%	0.60
100	2	74	0.2%	1.00
	4	127	0.7%	0.98
	8	174	1.8%	0.96
	16	217	4.3%	0.91
500	2	237	0.03%	1.00
	4	377	0.1%	1.00
	8	453	0.2%	1.00
	16	516	0.5%	0.88

5.3. Parallel efficiency with increasing number of second-stage constraints. For the second set of experiments, we fix the parameters $N = 128$, $m_0 = 100$, $n_0 = 100$, $n_1 = 1000$, and vary $m_1 = 100$, the number of second stage constraints.

The communication time T_{comm} varies linearly with m_1. As m_1 decreases, the effect of communication time grows. Compare the entries of Tables 2 and 3 showing ratios of 2.8%. At the Table 2 entry, $m_1 = m_0 = n_0 = 100$, eight scenarios are supplied to each processor, and the efficiency is 91%. At the Table 3 entry, $m_1 = 10$, $m_0 = n_0 = 100$, 64 scenarios are supplied to each processor, but the efficiency is only 88%. Because the fraction of time spent in communication is the same for both entries, the efficiency drop is attributed to a relative increase in the time for serial computation T_{serial}. In particular, when m_1 shrinks in relation to n_0, the relative

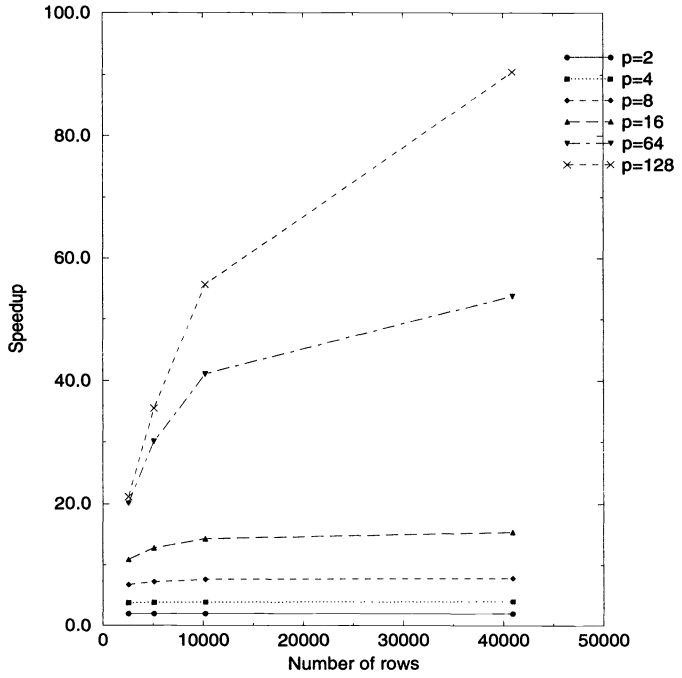


FIG. 1. *Speedups with the solution of the* SCSD8 *test problems on an* iPSC/860 *with up to* 128 *nodes.*

TABLE 4
Characteristics of the large-scale, real-world test problems.

Problem	Scenarios	Rows of A	Columns of A
SCSD8.16	16	330	2,310
SCSD8.64	64	1,290	9,030
SCSD8.256	256	5,130	35,910
SCSD8.512	512	10,250	71,750
SCSD8.2048	2,048	40,970	286,790
SCSD8.49152	49,152	983,050	6,881,350
SCSD8.98304	98,304	1,966,090	13,762,630
SEN.16	16	2,801	12,810
SEN.64	64	11,201	50,970
SEN.256	256	44,801	203,610
SEN.512	512	89,601	407,130
SEN.6144	6,144	1,075,201	4,884,570

contribution of T_{serial} (largely the dense $n_0 \times n_0$ system solutions) is greater [6].

Thus, all else being equal, the second set of experiments shows that the best efficiency of the parallel implementation is achieved when the ratio n_0/m_1 is as small as possible. Experiments with fixed m_1 and varying n_0 confirm this result [6].

6. Portability of the code: implementation on the Connection Machine CM–5. The modular structure of the code makes it portable to other machines suitable for coarse-grained parallel tasks. In this section, we show how we ported our code to a second distributed-memory MIMD computer—the Connection Machine CM–5. The processors of the CM–5 are connected according to a *fat-tree* network, and

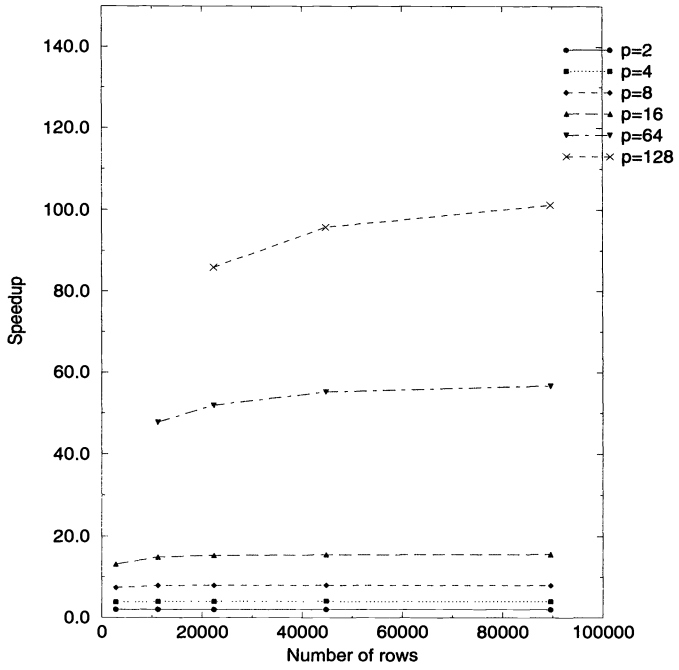


FIG. 2. *Speedups with the solution of the* SEN *test problems on an* iPSC/860 *with up to* 128 *nodes.*

near-optimal communication algorithms (such as broadcast) based on pseudorandom scheduling of the interprocessor links have been devised [8]. Due to the complexity of these algorithms, we have not written our own implementations but rather have used those provided in the CM–5 communication libraries. Using these routines made the port to the CM–5 straightforward. However, we also made a simple modification to the iPSC/860 algorithm to improve its parallel efficiency on the CM–5.

The modification deals with the global summation required to form G_1 and $\hat{p}_1$ at steps 2(a) and 2(b). In the implementation on the iPSC/860, G_1 and $\hat{p}_1$ are formed on node 0 and steps 2(c)–2(e) are carried out serially on node 0. The vector q_1 produced at step 2(e) is then broadcast to all nodes before step 3. On the Connection Machine CM–5, the `CMMD-reduce-v` global vector reduction function takes, as input, vectors (or matrices) distributed in every processor, sums them, and leaves a single vector (or matrix) sum at every node. Using this routine we can implement `Finddy` such that steps 2(c)–2(e) are performed redundantly on all nodes. Hence, every processor has a copy of vector q_1 (at the end of step 2(e)), and the broadcast call (STB) before step 3 is no longer necessary.

The modified parallel `Finddy` on the CM–5, with the redundant calculations, is summarized below.

PROCEDURE 6.1 (Parallel `Finddy` CM5($S, A_0, T_1, \ldots, T_N, b, dy$)). *Begin with the following data distribution: Node* l *holds* S_{l+1}, T_{l+1} *and* b_{l+1}, $l = 0, \ldots, N-1$. *Each node also holds* A_0, S_0, D_0, *and* b_0.

1. (*In parallel, solve* $Sp = b$). *On all nodes (redundantly) solve* $S_0 p_0 = b_0$. *In parallel (on nodes* $l = 0, \ldots, N-1$), *solve* $S_{l+1} p_{l+1} = b_{l+1}$ *for* p_{l+1}.

2. (*Solve* $Gq = V^\top p$).

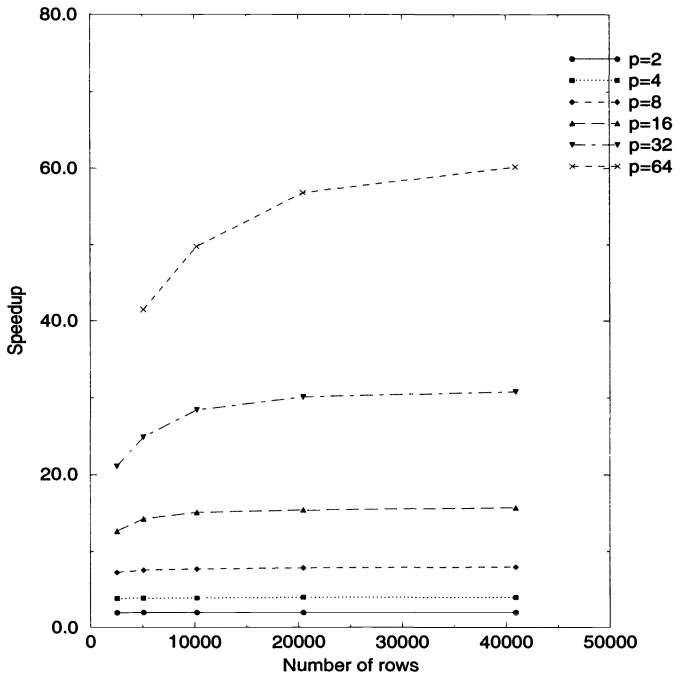


FIG. 3. *Speedups with the solution of the* SCSD8 *test problems on a Connection Machine* CM–5 *with up to* 64 *nodes.*

(a) *In parallel on nodes* $l = 0, \ldots, N-1$ *solve* $S_{l+1}(u_{l+1})_i = (T_{l+1})_{\cdot i}$ *for* $(u_{l+1})_i$, $i = 1, \ldots, n_0$.
(b) *In parallel on nodes* $l = 0, \ldots, N-1$ *multiply* $T_{l+1}^\top p_{l+1}$.
On all nodes (redundantly) compute $\hat{p}_2$.
Call `CMMD-reduce-v` *to form* G_1 *and* $\hat{p}_1$ *on all nodes* $l = 0, \ldots, N-1$.
(c) *On all nodes (redundantly) solve* $G_1 u = \hat{p}_1$ *for* u *and set* $v = \hat{p}_2 + A_0 u$.
(d) *On all nodes (redundantly) form* G_2 *by solving* $(G_1)w_i = (A_0^\top)_{\cdot i}$ *for* w_i, *for* $i = 1, \ldots, m_0$ *and setting* $G_2 = -A_0[w_1, \ldots, w_{m_0}]$.
(e) *On all nodes (redundantly) solve* $G_2 q_2 = -v$ *for* q_2 *and solve* $G_1 q_1 = \hat{p}_1 - A_0^\top q_2$ *for* q_1.

3. (*In parallel, solve* $Sr = Uq$). *On all nodes (redundantly) set* $r_0 = A_0 q_1 + q_2$. *On nodes* $l = 0, \ldots, N-1$, *solve* $S_{l+1} r_{l+1} = T_{l+1} q_{l+1}$ *for* r_{l+1}.

4. (*In parallel, form dy.*) *On all nodes (redundantly) set* $dy_0 = p_0 - r_0$. *On nodes* $l = 0, \ldots, N-1$, *set* $(dy)_{l+1} = p_{l+1} - r_{l+1}$.

7. Solving large-scale problems. We apply now both the iPSC/860 and the CM–5 codes to factor and solve the systems of equations arising in two sets of real-world test problems. The SCSD8 set is a stochastic version of a model to find the minimal design of a multistage truss [3]. SEN is a telecommunication network design problem communicated to us by Suvrajeet Sen from University of Arizona. SCSD8 has dimensions $m_0 = 10$, $n_0 = 70$, $m_1 = 20$, $n_1 = 140$. SEN has dimensions $m_0 = 1$, $n_0 = 90$, $m_1 = 175$, $n_1 = 795$. We generated problems by increasing the number of scenarios. Table 4 summarizes the characteristics of the test problems. To put the size of these problems in perspective we mention that Birge and Holmes [3] solved SCSD8 with only 32 scenarios. To the best of our knowledge the largest SEN problem

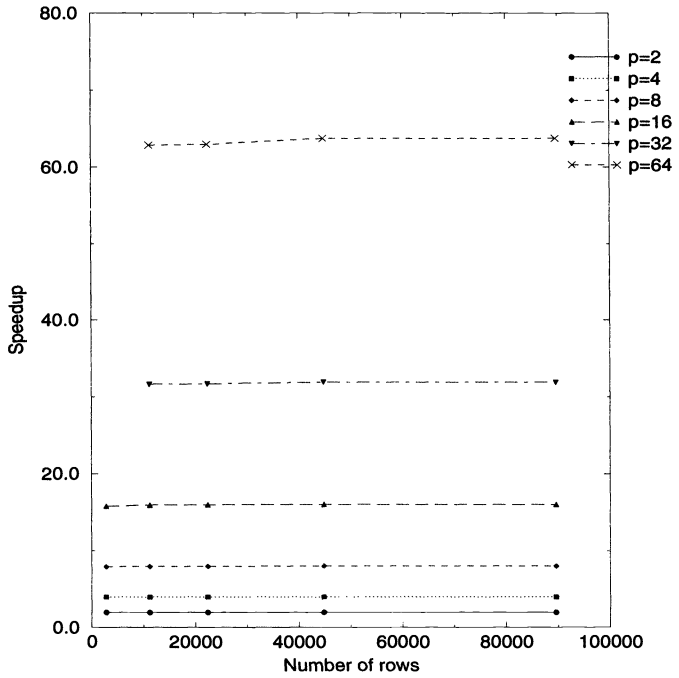


FIG. 4. *Speedups with the solution of the* SEN *test problems on a Connection Machine* CM–5 *with up to* 64 *nodes.*

solved to date has 100 scenarios.

We use the implementations on both the iPSC/860 and the CM–5 to solve the linear systems arising from the set of problems summarized in Table 4. These experiments illustrate the efficiency of the factorization routines in solving extremely large problems. They also serve as a benchmark comparison of the iPSC/860 and the CM–5 implementations.

Figures 1 and 2 illustrate the speedups achieved when solving the SCSD8 and SEN sets, respectively, on an iPSC/860 with up to 128 nodes. For cube sizes up to 16 nodes, the speedup curves quickly level off at $S_p \approx p$. That is, near perfect speedup is achieved for most matrix dimensions. For the 128-node cube size, the number of scenarios per processor N/p that fits into processor memory is insufficient to provide better than 70% efficiency for the largest SCSD8 problem and 79% efficiency for the largest SEN problem. However, the 128-node plots do indicate an increasing trend of speedup versus matrix dimension even for the largest matrices solved.

Figures 3 and 4 illustrate the speedups achieved when solving the SCSD8 and SEN sets on a CM–5 with up to 64 nodes. The efficiency of the parallel implementation is more than 99% for the larger matrices. Very good efficiency is achieved even for the smaller test problems.

Table 5 summarizes the solution times for the real-world test problems on the iPSC/860, the Connection Machine CM–5, and a prototype "Viking" system, which is the precursor of the CM–5e. (The CM–5 is based on SPARC 2 processors, and the CM–5e is based on SPARC 10 stations. Improvements in performance are expected both in the processing power and communication efficiency of the production CM–5e compared to the prototype "Viking.") Both the CM–5 and the "Viking" CM–5

TABLE 5

Solution times (in seconds) for the large-scale, real-world test problems on the Intel iPSC/860, *the Connection Machine* CM–5, *and the prototype "Viking" Connection Machine* CM–5. NR = *did not run, due to limited computing resources.* NEM = *not enough memory on the largest machine available to us.*

Problem	Intel		CM–5		"Viking" CM–5	
	Nodes	Time	Nodes	Time	Nodes	Time
SCSD8.16	16	0.144	16	0.157	16	0.073
SCSD8.64	16	0.228	16	0.332	16	0.131
SCSD8.256	16	0.572	16	1.035	16	0.364
SCSD8.512	16	1.031	16	2.013	16	0.668
SCSD8.2048	16	3.820	16	7.606	16	2.530
SCSD8.49152	NR	NR	64	46.61	64	14.88
SCSD8.98304	128	23.650	64	NEM	64	NEM
SEN.16	16	0.970	16	1.635	16	0.739
SEN.32	32	0.988	32	1.635	32	0.739
SEN.64	64	1.043	64	1.636	64	0.741
SEN.128	128	1.157	64	3.403	64	1.549
SEN.256	128	2.077	64	7.054	64	3.156
SEN.512	128	3.929	64	14.390	64	6.332
SEN.6144	NR	NR	64	171.330	64	76.540

processors have vector functional units, but the current version of our code does not use them.

Note that for the SCSD8 test problems, the speedups are nearly identical on the iPSC/860 and the CM–5. This is also the case for the SEN test problems when N/p is large enough. The use of redundant computation in the CM–5 implementation reduces the effects of T_{comm} on the speedup when N/p is small. For both machines, the SEN problem (with $n_0/m_1 = 0.5$) is more efficient at small N/p than is the SCSD8 problem (with $n_0/m_1 = 3.5$). This confirms our observation that good parallel performance is achieved for problems with small n_0/m_1 ratio.

8. Conclusions. We have shown in this paper that the BQ matrix factorization procedure can achieve excellent speedups on distributed-memory multiprocessors (like the Intel iPSC/860 and the Connection Machine CM–5). These speedups are achieved with the development of efficient communication schemes that have been adapted to the needs of the BQ algorithm.

The results in Table 5 illustrate the excellent scalability of the parallel implementation. On all machines, the solution time for the SEN problems with 16, 32, and 64 scenarios remains virtually constant as we use 16, 32, and 64 processors, respectively.

Using these matrix factorization routines within an interior point algorithm, we can take advantage of the fact that structurally identical matrices are factored and solved at each iteration of the algorithm. Hence, the symbolic factorization of the matrices can be computed once and used in all iterations. This technique—standard in interior point codes—will reduce the solution time per iteration, but may also cause some degradation in speedup. The use of parallel `Finddy` within an interior point code, extensions to the multistage model, and use of the vector functional units on the CM–5 are topics of our current work.

Acknowledgments. The authors wish to thank E. Ng and B. Peyton for providing their codes and information about their use. Experiments were performed on the iPSC/860 machine at Oak Ridge National Laboratory and on the CM–5 Thinking Machines Corporation and at the AHPCRC, University of Minnesota.

REFERENCES

[1] E. ANDERSON, Z. BAI, C. BISCHOF, J. DEMMEL, J. DONGARRA, J. DUCROZ, A. GREENBAUM, AND S. HAMMARLING, *LAPACK User's Guide*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1992.

[2] J. R. BIRGE AND L. QI, *Computing block-angular Karmarkar projections with applications to stochastic programming*, Management Sci., 34 (1988), pp. 1472–1479.

[3] J. R. BIRGE AND D. F. HOLMES, *Efficient solution of two-stage stochastic linear programs using interior point methods*, Comput. Optim. Appl., 1 (1992), pp. 245–276.

[4] I. CHOI AND D. GOLDFARB, *Solving multicommodity network flow problems by an interior point method*, in Large Scale Numerical Optimization, T. Coleman and Y. Li, eds., Society for Industrial and Applied Mathematics, Philadelphia, PA, 1990, pp. 58–69.

[5] J. J. DONGARRA, J. DU CROZ, I. DUFF, AND S. HAMMARLING, *A set of level 3 basic linear algebra subroutines*, Preprint No. 2, Argonne National Laboratory, Agonne, IL, 1988.

[6] E. R. JESSUP, D. YANG, AND S. A ZENIOS, *Parallel factorization on the* iPSC/860 *of structured matrices arising in stochastic programming*, Tech. Report CU-CS-701-94, Dept. of Computer Science, University of Colorado at Boulder, 1994.

[7] P. KALL AND E. KELLER, GENSLP: *A program for generating input for stochastic linear programs with complete fixed recourse*, Institute for Operations Research der University Zurich, Zurich CH-8006, Switzerland, 1985, manuscript.

[8] C. E. LEISERSON, Z. S. ABUHAMDEH, D. C. DOUGLAS, C. R. FEYNMAN, M. N. GANMUKHI, J. V. HILL, W. D. HILLIS, B. C. KUSZMAUL, M. A. ST. PIERRE, D. S. WELLS, M. C. WONG, S.-W. YANG, AND R. ZAK, *The Network Architecture of the Connection Machine* CM–5, Thinking Machines Corporation, Cambridge, MA 02142, 1992, manuscript.

[9] I. J. LUSTIG, J. M. MULVEY, AND T. J. CARPENTER, *Formulating two-stage stochastic programs for interior point methods*, Oper. Res., 39 (1991), pp. 757–770.

[10] J. M. MULVEY AND A. RUSZCZYNSKI, *A diagonal quadratic approximation method for large scale linear programs*, Oper. Res. Lett., 12 (1992), pp. 205–215.

[11] E. NG AND B. PEYTON, *A supernodal Cholesky factorization algorithm for shared-memory multiprocessors*, SIAM J. Sci. Comput., 14 (1993), pp. 761–769.

[12] S. NIELSEN AND S.A. ZENIOS, *A massively parallel algorithm for nonlinear stochastic network problems*, Oper. Res., 41 (1993), pp. 319–337.

[13] R. T. ROCKAFELLAR AND R. J.-B. WETS, *Scenarios and policy aggregation in optimization under uncertainty*, Math. Oper. Res., 16 (1991), pp. 119–147.

[14] Y. SAAD AND M. H. SCHULTZ, *Data communication in hypercubes*, J. Parallel Distributed Computing, 6 (1989), p. 115ff.

[15] R. J.-B. WETS, *Stochastic programs with fixed resources: the equivalent deterministic problem*, SIAM Rev., 16 (1974), pp. 309–339.

[16] S.A. ZENIOS, *A model for portfolio management with mortgage-backed securities*, Ann. Oper. Res., 43 (1993), pp. 337–356.

SIAM J. Optimization
Vol. 4, No. 4, pp. 847–871, November 1994
© 1994 Society for Industrial and Applied Mathematics
010

STRUCTURED LINEAR LEAST-SQUARES PROBLEMS IN SYSTEM IDENTIFICATION AND SEPARABLE NONLINEAR DATA FITTING*

LINDA KAUFMAN†, GARRETT S. SYLVESTER‡, AND MARGARET H. WRIGHT§

Abstract. Linear least-squares problems involving very large, completely dense matrices with a highly special structure arise in two applications—system identification and separable nonlinear least-squares with multiple data sets. Generic dense methods applied to such problems would be inordinately costly in terms of both speed and storage. Various techniques for exploiting the structure are described that differ in operation count, storage requirements, accuracy, and suitability for parallel computation. The system identification problem is presented first, to characterize the structure and analyze the algorithmic alternatives. Some indication is then given of how to apply the same techniques to separable nonlinear least-squares with multiple data sets.

Key words. system identification, matrix structure, linear least-squares, normal equations, QR factorization, separable nonlinear least-squares

AMS subject classifications. 65F20, 65F25, 65K10, 90C06, 93A15, 93E24

1. Introduction.

Many discussions of large-scale problems in linear algebra and optimization tend to assume, explicitly or implicitly, that all relevant matrices are *sparse*. Given a problem that is very large but completely dense, it might appear at first that the only option is to use "black-box" dense techniques. This paper addresses the situation in which, although a problem is both large and dense, it exhibits a high degree of *structure* that can be exploited to speed up computation and reduce storage requirements. We focus on the structured linear least-squares problems arising in two different contexts: system identification and separable nonlinear least-squares with multiple data sets.

1.1. System identification.

System identification involves constructing mathematical models of dynamic systems based on experimental data. Specialized numerical techniques have been developed for a variety of system identification problems, mostly concentrating on the time domain. (For background on system identification and the associated numerical methods, see, for example, [14], [16], and the references therein.) In contrast, the problem of interest in this paper is *frequency-domain* system identification: the estimation of modal parameters from multiple-driver, multiple-receiver transfer function data. Several approaches to various frequency-domain problems are given in [2], [3], [5], [13], [15], [17], [18]. Although our methods have been developed for frequency-domain identification, we believe that they can also be applied in the time domain.

The transfer functions typically describe the vibrations of structurally complex objects and contain many modes. Issues of numerical accuracy are central because the modes have two scales: extremely narrow resonances centered in the band where the data are measured; and others lying outside this band, but including broad diffuse features that "drape" into the analysis band. An overall aim is to determine the resonant frequencies in the band of interest. However, it is difficult to make a numerical distinction between the two classes of modes based on the observed data.

* Received by the editors August 23, 1993; accepted for publication (in revised form) March 22, 1994.

† AT&T Bell Laboratories, Murray Hill, New Jersey 07974 (lck@research.att.com).
‡ AT&T Bell Laboratories, Whippany, New Jersey 07981 (gss@hogpa.ho.att.com).
§ AT&T Bell Laboratories, Murray Hill, New Jersey 07974 (mhw@research.att.com).

One of the numerical subproblems in these applications includes the following class of large, dense linear least-squares problems: given two complex matrices G and J, both $m \times n$, a complex matrix H of dimension $r \times n$, and a complex right-hand side b, find the complex n-vector z that solves the linear least-squares problem

$$\text{(1.1)} \qquad \text{minimize} \quad \sum_{i=1}^{m} \sum_{j=1}^{r} \left| \left(\sum_{k=1}^{n} G_{ik} H_{jk} z_k + J_{ik} H_{jk}^* z_k^* \right) - b_{ij} \right|^2,$$

where the superscript "$*$" denotes the complex conjugate. The problem (1.1) must be solved for d right-hand side vectors.

We refer throughout to a "typical" problem that was central in a project of the second author. For this problem, $m = 500$ (number of rows of G and J), $r = 200$ (number of rows of H), $d = 20$ (number of right-hand sides), and $n = 60$ (number of unknowns per right-hand side). Note that, for this example, there are many more rows $(100,000)$ than columns (60), and that the number of right-hand sides is comparable to (although smaller than) the number of unknowns. Beyond their large size and special structure, these problems have two properties of interest: they are often seriously ill-conditioned because of the difficulties in identification mentioned above, and the optimal residual is quite small because the model fits the data closely.

1.2. Separable nonlinear least-squares with multiple data sets. In least-squares data fitting, parameters are chosen to minimize the two-norm of the difference between observations and model predictions. Our focus is on *separable* nonlinear least-squares problems with multiple sets of observations where the same nonlinear parameters appear for all data sets ("SNLSMD" problems). For example, a scientist may wish to fit the model

$$c_1 \exp(x_1 t_i) + c_2 \exp(x_2 t_i)$$

to 100 sets of experimental data using the same values of x_1 and x_2, but with coefficients c_1 and c_2 that depend on the experiment.

Most nonlinear least-squares solvers pose a sequence of linear least-squares subproblems. For some formulations of SNLSMD problems, the subproblem matrices have a structure similar to matrices in the system identification problem, so that all the machinery developed for one problem class can also be applied to the other. We return to nonlinear least-squares in §10 after discussing the system identification problem.

2. A simplified real problem. The linear algebraic issues arising in solving (1.1) are explained most easily by focusing on a simplified problem involving only real quantities. Given an $m \times n$ matrix G, an $r \times n$ matrix H (both of full rank), and a right-hand side b with mr components, we wish to find the unknown n-vector x that solves the least-squares problem

$$\text{(2.1)} \qquad \text{minimize} \quad \sum_{i=1}^{m} \sum_{j=1}^{r} \left(\sum_{k=1}^{n} g_{ik} h_{jk} x_k - b_{ij} \right)^2,$$

where g_{ik} denotes the (i,k) element of G, and so on. In effect, the right-hand side is being synthesized from the rank-one matrices $\{g_{ik} h_{jk}\}$ by means of the coefficients $\{x_k\}$; the special structure results because the basis elements have rank one.

Much of our discussion will involve a single generic right-hand side, which is denoted by the mr-vector b. It is sometimes convenient to group the elements of b

into r subvectors of dimension m:

$$b = \begin{pmatrix} b^{(1)} \\ \vdots \\ b^{(r)} \end{pmatrix}. \tag{2.2}$$

In discussing multiple right-hand sides, we use B to represent the $mr \times d$ matrix whose jth column is the jth right-hand side, denoted in this context by b_j: $B = (\, b_1 \quad b_2 \quad \cdots \quad b_d \,)$.

In matrix form, the problem (2.1) is to minimize $\|Ax-b\|_2^2$, where the least-squares matrix A is given by

$$A = \begin{pmatrix} g_{11}h_{11} & g_{12}h_{12} & g_{13}h_{13} & \cdots & g_{1n}h_{1n} \\ \vdots & \vdots & \vdots & & \vdots \\ g_{m1}h_{11} & g_{m2}h_{12} & g_{m3}h_{13} & & g_{mn}h_{1n} \\ & & \vdots & & \\ g_{11}h_{r1} & g_{12}h_{r2} & \cdots & \cdots & g_{1n}h_{rn} \\ \vdots & \vdots & & & \vdots \\ g_{m1}h_{r1} & g_{m2}h_{r2} & \cdots & & g_{mn}h_{rn} \end{pmatrix}. \tag{2.3}$$

Note that A is a stack of r submatrices, each with m rows; the ith submatrix is the matrix G with its n columns scaled by the elements of the ith row of H. Defining $H^{(i)}$ as the $n \times n$ diagonal matrix whose nonzero elements are those of the ith row of H, so that $H^{(i)} = \operatorname{diag}(h_{i1}, \ldots, h_{in})$, the matrix A of (2.3) can be written as

$$A = \begin{pmatrix} GH^{(1)} \\ \vdots \\ GH^{(r)} \end{pmatrix}. \tag{2.4}$$

Because $m > r$ in our typical problem, the form (2.4) leads to greater efficiency in the methods described in §4. Obviously the least-squares matrix can equally well be row-permuted and written in terms of diagonal matrices $\{G^{(i)}\}$.

The matrix A of (2.4) has mr rows and n columns. (For the typical problem mentioned in §1.1, A has 100,000 rows and 60 columns.) Furthermore, although A is in general *completely dense*, it is *highly structured*. Despite the lack of sparsity in (2.4), the associated least-squares problem can be solved efficiently using a variety of techniques based on its structure.

3. The normal equations. The most obvious method for solving the linear least-squares problem is to use the classical *normal equations* $A^TAx = A^Tb$. The normal equations are discussed in essentially all linear algebra textbooks; see, for example, [9]. Since A^TA is symmetric and positive semidefinite, the normal equations are typically solved using the upper-triangular Cholesky factor R_N of the normal-equation matrix A^TA. Calculation of the Cholesky factor R_N of the $n \times n$ matrix A^TA requires approximately $n^3/6$ multiplications (and a similar number of additions). Because n in the problems of interest is not very large (no more than a few hundred), a standard dense Cholesky factorization would be the method of choice.

If A is treated as a general dense matrix, the amount of work needed to form the normal-equation matrix is approximately $\frac{1}{2}mrn^2$ multiplications. But we know from the special structure (2.4) of A that

$$A^TA = H^{(1)}G^TGH^{(1)} + H^{(2)}G^TGH^{(2)} + \cdots + H^{(r)}G^TGH^{(r)}. \tag{3.1}$$

It follows from the definition of $H^{(i)}$ that the (i,j) element of A^TA is given by

$$(A^TA)_{ij} = (G^TG)_{ij}\ (H^TH)_{ij}. \tag{3.2}$$

The matrix A^TA can consequently be obtained by separately forming the two explicit $n \times n$ matrices G^TG and H^TH, and then multiplying together their (i,j) elements, using symmetry in the obvious way. The total work required to form A^TA using (3.1) is approximately $\frac{1}{2}(m+r)n^2$ multiplications. For the typical values $m = 500$ and $r = 200$, forming A^TA using the structure of A requires about $350n^2$ multiplications, compared to $50,000n^2$ multiplications if structure is not exploited.

Unfortunately, because the right-hand side b has no special structure, forming A^Tb requires the same effort (mnr multiplications) as if A were a general matrix. Table 6.1 (see §6) shows that, if $m > r > n$, even with only one right-hand side, the most expensive single calculation for the structured normal-equation method is the formation of A^Tb. The structured normal-equation method needs to store G, H, and a single copy of the right-hand side; see Table 6.3.

The structured normal-equation approach is extremely efficient in terms of work—in fact, it is the most efficient of all approaches considered in this paper—but it may be numerically unreliable for the ill-conditioned small-residual problems of interest here. The next two sections describe possible alternatives.

4. Structured orthogonal triangularization. For dense least-squares problems, a standard numerically stable alternative to the normal equations involves an orthogonal-triangular factorization of A. The QR factorization, for example, produces an orthogonal matrix Q_A and an $n \times n$ upper-triangular matrix R_A such that

$$Q_AA = \begin{pmatrix} R_A \\ 0 \end{pmatrix}. \tag{4.1}$$

The least-squares solution is found by solving

$$R_Ax = b_A, \quad \text{where } b_A = \text{first } n \text{ components of } Q_Ab.$$

For general dense problems, using the QR factorization takes approximately twice as much work as the normal equations. Three methods will be presented that are designed to exploit the structure of A through orthogonal triangularization.

4.1. Struct1. The special structure (2.4) of A contains submatrices in which G appears on the left. We now describe a two-step process ("struct1") for reducing A to upper-triangular form. The first step relies on the QR factorization of G:

$$QG = \tilde{R} = \begin{pmatrix} R \\ 0 \end{pmatrix}, \tag{4.2}$$

where Q is an $m \times m$ orthogonal matrix, usually constructed from a sequence of Householder transformations, and R is an $n \times n$ upper triangle. (The form (4.2)

assumes that $m \geq n$.) If we (conceptually) apply the mr-dimensional block-diagonal orthogonal matrix $\tilde{Q} = \text{diag}(Q, Q, \ldots, Q)$ to A, the result is

$$\tilde{Q}A = \begin{pmatrix} Q & & \\ & \ddots & \\ & & Q \end{pmatrix} \begin{pmatrix} GH^{(1)} \\ \vdots \\ GH^{(r)} \end{pmatrix} = \begin{pmatrix} \tilde{R}H^{(1)} \\ \vdots \\ \tilde{R}H^{(r)} \end{pmatrix}. \tag{4.3}$$

(A more compact way to express this process is as the application of the tensor product $I \otimes Q$ to A.) The transformed matrix $\tilde{Q}A$ is a stack of r upper triangles, each $n \times n$, separated from one another by r blocks of $m - n$ zero rows; the ith upper triangle is the matrix R from (4.2) with its columns scaled by the diagonal matrix $H^{(i)}$.

An important property of least-squares problems is that zero rows appearing in the least-squares matrix do not affect the solution and can therefore be "deleted" from the problem. Let $\tilde{A}$ denote the matrix $\tilde{Q}A$ of (4.3) with the zero rows removed, so that $\tilde{A}$ has nr rows and n columns:

$$\tilde{A} = \begin{pmatrix} RH^{(1)} \\ \vdots \\ RH^{(r)} \end{pmatrix}. \tag{4.4}$$

When $r = 4$ and $n = 3$, the form of $\tilde{A}$ is shown in Fig. 4.1, where "$\times$" denotes a generic nonzero element.

$$\tilde{A} = \begin{pmatrix} \times & \times & \times \\ & \times & \times \\ & & \times \\ \times & \times & \times \\ & \times & \times \\ & & \times \\ \times & \times & \times \\ & \times & \times \\ & & \times \\ \times & \times & \times \\ & \times & \times \\ & & \times \end{pmatrix}.$$

FIG. 4.1. *Form of* $\tilde{A}$.

To complete the solution process, $\tilde{A}$ must be reduced to triangular form. This reduction can be accomplished by a special sequence of n Householder transformations in which the jth transformation is designed to annihilate the $j(r-1)$ nonzero subdiagonal elements in column j; see Fig. 4.1. No fill-in occurs when the columns are reduced from left to right. Letting Q_1 denote the product of the n special Householder transformations, we have

$$Q_1\tilde{A} = \begin{pmatrix} R_A \\ 0 \end{pmatrix}. \tag{4.5}$$

Three steps are required to obtain the least-squares solution. First, $\tilde{Q}$ is applied to b, which produces an mr-vector divided into r subvectors, of which the ith is $Qb^{(i)}$; see (2.2). Let $\tilde{b}$ denote the nr-vector defined by the elements of $\tilde{Q}b$ corresponding to nonzero rows of $\tilde{Q}A$. Second, Q_1, the special sequence of Householder transformations

indicated in (4.5), is applied to $\tilde{b}$. Finally, the triangular system $R_A x = b_A$ is solved for x, where b_A is the first n components of $Q_1 \tilde{b}$.

In summary, the steps of struct1 are:

Procedure Struct1:

1. Calculate $QG = R$; see (4.2). Construct $\tilde{A}$ of (4.4).
2. Define Q_1 and calculate R_A; see (4.5).
3. Apply $\tilde{Q}$ to b, and form $\tilde{b}$. Apply Q_1 to $\tilde{b}$ and construct b_A.
4. Solve $R_A x = b_A$.

Operation counts for struct1 are given in Table 6.1. Compared to the structured normal equations, struct1 suffers a ten-fold increase in work in processing A, with double the work for the right-hand side.

Table 6.3 summarizes the storage requirements for struct1. The matrix G can be overwritten with R and the transformations comprising Q. Application of Q_1 can be implemented in two ways, which involve the same work but differ in storage requirements; the choice depends on the relative sizes of n and d.

1. Version 1 of struct1. The Householder transformations that compose Q_1 are stored as a set of vectors, taking advantage of their known zero structure; see Fig. 4.1.
2. Version 2 of struct1. The transformations comprising Q_1 are not saved, but rather applied as they are generated to all right-hand sides. For each right-hand side, the nr components remaining after application of $\tilde{Q}$ are saved.

4.2. Struct2 and struct3. The matrix $\tilde{A}$ produced by applying $\tilde{Q}$ to A and deleting the zero rows may be written out in detail as:

$$\tilde{A} = \begin{pmatrix} r_{11}h_{11} & \cdots & r_{1n}h_{1n} \\ & \ddots & \vdots \\ & & r_{nn}h_{1n} \\ & \vdots & \\ r_{11}h_{r1} & \cdots & r_{1n}h_{rn} \\ & \ddots & \vdots \\ & & r_{nn}h_{rn} \end{pmatrix}. \tag{4.6}$$

Suppose that we rearrange the nr rows of $\tilde{A}$ by first grouping together every nth row, starting with row 1; then grouping together every nth row starting with row 2; and so on. The resulting matrix, denoted by $\bar{A}$, is a stack of n blocks, each with r rows. The ith block is given by $HR^{(i)}$, where $R^{(i)}$ denotes the $n \times n$ diagonal matrix created from the ith row of R.

Let P_R denote the permutation that creates the desired rearrangement, so that

$$\bar{A} = P_R \tilde{A} = \begin{pmatrix} HR^{(1)} \\ \vdots \\ HR^{(n)} \end{pmatrix}. \tag{4.7}$$

The $r \times n$ matrix H appears on the left in each block of $\bar{A}$, and is therefore in a position to be transformed from the left. Since R is upper-triangular, $R^{(1)}$ contains n nonzeros, $R^{(2)}$ contains $n-1$ nonzeros, and so on. The special structure of $R^{(i)}$ implies that, moving down the blocks, each successive block of $\bar{A}$ has one additional

zero column at the front. The next two subsections describe two straightforward ways to take advantage of the special structure of $\bar{A}$.

4.2.1. Struct2. The first such approach ("struct2") begins with the QR factorization of H:

$$YH = \tilde{T} = \begin{pmatrix} T \\ 0 \end{pmatrix}, \tag{4.8}$$

where Y is orthogonal and T is an $n \times n$ upper triangle. (Note the assumption that $r > n$.) If the $nr \times nr$ orthogonal matrix $\tilde{Y} = \operatorname{diag}(Y, Y, \ldots, Y)$ is applied to $\bar{A}$ of (4.7), the resulting transformed matrix contains n upper triangles, each $n \times n$, separated by groups of $r - n$ zero rows. After removing these zero rows, we have a matrix, denoted by $\hat{A}$, that is a stack of n upper triangles of dimension $n \times n$, with an additional zero column at the front of each block as we move down the stack:

$$\hat{A} = \begin{pmatrix} TR^{(1)} \\ \vdots \\ TR^{(n)} \end{pmatrix}. \tag{4.9}$$

For $n = 4$, $\hat{A}$ has the form shown in Fig. 4.2.

$$\hat{A} = \begin{pmatrix} \times & \times & \times & \times \\ & \times & \times & \times \\ & & \times & \times \\ & & & \times \\ & \times & \times & \times \\ & \times & \times & \times \\ & & \times & \times \\ & & & \times \\ & & \times & \times \\ & & \times & \times \\ & & & \times \\ & & & \times \\ & & & \times \\ & & & \times \\ & & & \times \\ & & & \times \end{pmatrix}$$

FIG. 4.2. *Form of* $\hat{A}$.

The matrix $\hat{A}$ contains approximately $\frac{1}{3}n^3$ nonzero subdiagonal elements that need to be annihilated. As with the structure $\tilde{A}$ of Fig. 4.1, a special sequence of n Householder transformations, applied to the columns in natural order, can be constructed to reduce $\hat{A}$ to upper-triangular form without intermediate fill-in. Let Q_2 denote the product of these transformations, so that

$$Q_2 \hat{A} = \begin{pmatrix} R_A \\ 0 \end{pmatrix}. \tag{4.10}$$

To complete struct2, the right-hand side needs to be transformed by the three orthogonal matrices that produce R_A. Hence we first apply $\tilde{Q}$ to b, producing $\tilde{b}$; second, $\tilde{Y}$ is applied to a permuted version of $\tilde{b}$, producing $\hat{b}$; third, Q_2 is applied to $\hat{b}$, producing $\bar{b}$. Finally, the least-squares solution is obtained by solving $R_A x = b_A$, where b_A denotes the first n components of $\bar{b}$.

4.2.2. Struct3. The third and final approach ("struct3"), which is closely related to struct2, is suggested by the observation that, in the matrix $\bar{A}$ of (4.7), post-multiplication by the diagonal matrices $R^{(i)}$ has the effect of removing an additional column from the front of each successive block. This property suggests reducing H not to an upper triangle as in (4.8), but instead to a form with more nonzero elements in the early columns—for example, a lower triangle. We thus calculate the "QL" factorization of H:

$$VH = \tilde{L} = \begin{pmatrix} L \\ 0 \end{pmatrix}, \tag{4.11}$$

where V is orthogonal and L is an $n \times n$ lower triangle.

Multiplying $\bar{A}$ of (4.7) by the orthogonal matrix $\tilde{V} = \text{diag}(V, V, \ldots, V)$, we obtain a stack of matrices of the form $LR^{(i)}$, for $i = 1, \ldots, n$. A key difference from the form $\hat{A}$ of Fig. 4.2 arises because L is lower-triangular. This means that the top of each successive matrix $LR^{(i)}$ contains $i - 1$ additional zero rows.

After removing these additional zero rows from $\tilde{V}\bar{A}$, we have a matrix, denoted by $\breve{A}$, that is a stack of n shrinking and shifting lower triangles. ("Shrinking" means that in each successive triangle the number of rows decreases by one.) When $n = 5$, for example, $\breve{A}$ has the form shown in Fig. 4.3.

$$\breve{A} = \begin{pmatrix}
\times & & & & \\
\times & \times & & & \\
\times & \times & \times & & \\
\times & \times & \times & \times & \\
\times & \times & \times & \times & \times \\
 & \times & & & \\
 & \times & \times & & \\
 & \times & \times & \times & \\
 & \times & \times & \times & \times \\
 & & \times & & \\
 & & \times & \times & \\
 & & \times & \times & \times \\
 & & & \times & \\
 & & & \times & \times \\
 & & & & \times
\end{pmatrix}.$$

FIG. 4.3. *Form of* $\breve{A}$.

The number of elements in $\breve{A}$ to be annihilated to obtain a lower-triangular form in the first n rows is approximately $\frac{1}{6}n^3$, half the number associated with the matrix $\hat{A}$ of struct2 (Fig. 4.2). In contrast to the reduction of $\hat{A}$, however, unacceptably large *fill-in* would occur if we simply applied a straightforward sequence of n Householder transformations to the columns of $\breve{A}$ in reverse order. Fortunately, a sequence of Householder transformations can be constructed that efficiently creates a lower triangle while incurring only limited fill-in. In particular, each lower triangle, beginning in the last row, is completely annihilated by the triangle above it before moving up to the next triangle. With this approach, only a single element of fill is created in processing each row.

To illustrate the form of the reduction, Fig. 4.4 displays the sequence of transformations needed to annihilate the bottom three triangles for the case $n = 4$. The symbol "x" denotes a nonzero element that is not altered; "m" denotes a nonzero element that is modified; " f " denotes a previously zero element that is filled in; " 0 "

denotes a previously nonzero element that is annihilated; and blank or " · "denotes a zero element that is unaltered. Each "frame" is numbered for later reference in §9.

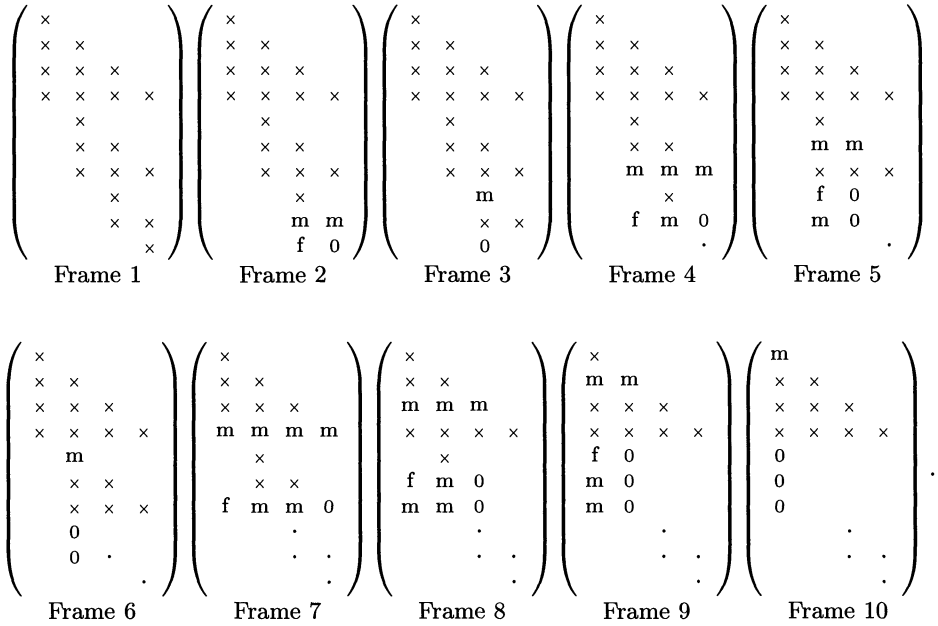


FIG. 4.4. *Reduction of* $\breve{A}$.

Let Q_3 denote the product of the transformations just described that reduce $\breve{A}$ to lower-triangular form, so that

$$Q_3 \breve{A} = \begin{pmatrix} L_A \\ 0 \end{pmatrix}, \tag{4.12}$$

where L_A is a lower-triangular matrix. Since $\breve{A}$ contains approximately half the number of elements as $\hat{A}$, the final step of the reduction in struct3 is half as costly as the final step in struct2. To complete struct3, the right-hand side needs to be transformed by the three orthogonal matrices that produce L_A.

4.2.3. Summary of struct2 and struct3. Procedures struct2 and struct3 are so similar that their steps are stated together.

Procedures Struct2 and Struct3:

1. Calculate $QG = R$ (see (4.2)); construct $\bar{A}$.
2. For struct2, calculate $YH = T$ (4.8) and construct $\hat{A}$ (Fig. 4.2). For struct3, calculate $VH = L$ (4.11), and construct $\breve{A}$ (Fig. 4.3).
3. For struct2, determine Q_2 and R_A; see (4.10). For struct3, determine Q_3 and L_A; see (4.12).
4. Apply $\tilde{Q}$ to b, and form $\tilde{b}$.
5. For struct2, apply $\tilde{Y}$ to a permuted version of $\tilde{b}$, construct $\hat{b}$, apply Q_2 to $\hat{b}$, and construct b_A. For struct3, apply $\tilde{V}$ to a permuted version of $\tilde{b}$, construct $\breve{b}$, apply Q_3 to $\breve{b}$, and construct b_A.
6. For struct2, solve $R_A x = b_A$. For struct3, solve $L_A x = b_A$.

Operation counts for struct2 and struct3 are given in Table 6.1. The amount of work needed to reduce A with struct2 and struct3 is less than with struct1, mainly because of the much smaller number of elements to be annihilated in the last step of the reduction. On the other hand, the work associated with the right-hand side for struct2 and struct3 is larger than for struct1, but not significantly so. Table 6.3 gives the storage required by struct2 and struct3 for approaches analogous to versions 1 and 2 of struct1.

5. The corrected seminormal equations. Instead of QR-based methods, we also considered using the method of corrected seminormal equations, as introduced and analyzed by Björck [4]. The original motivation for this method was to improve accuracy in solving linear least-squares problems by substituting the triangular matrix R_A from the QR factorization of A (4.1) into the normal equations, rather than using the Cholesky factor R_N of the matrix A^TA. With exact arithmetic, of course, R_A and R_N would be equal. For ill-conditioned least-squares problems, however, R_A should be, in some sense, "better" than R_N, since R_A comes from the more numerically stable QR factorization. The method of seminormal equations is defined as follows.

Seminormal equations (SNE):

1. Calculate R_A from the QR factorization of A.
2. Form A^Tb.
3. Solve the *seminormal equations* $R_A^TR_Ax_s = A^Tb$.

Despite the improved "quality" of R_A compared to R_N, the first and somewhat unexpected result from Björck's careful error analysis is that the seminormal-equation approach (SNE) is no more accurate than the normal equations (NE). Björck then defines the method of *corrected* seminormal equations (CSNE), in which a single step of iterative refinement is performed (without extended precision) after solving the seminormal equations. CSNE thus consists of SNE plus the following two steps:

4. Solve $R_A^TR_A\delta = A^T(b - Ax)$. This is called the *correction step.*
5. Define the corrected solution x_c as $x_s + \delta$.

Björck's second surprising (this time favorable) result is that, under mild conditions, the accuracy bounds for the solution produced by CSNE are comparable to those for a QR-based method. (For complete details, see [4].) The numerical results in §7 confirm the numerical reliability of CSNE.

The CSNE method was originally designed for problems in which A is large and sparse, where multiplying A or A^T by a vector is typically not expensive. In our system identification problem, however, although R_A (or, equivalently, L_A) can be calculated efficiently, applying A or A^T to an unstructured vector is *extremely expensive.* Compared to either the normal or seminormal equations, the extra work in CSNE associated with the right-hand side arises from the vector $A^T(b - Ax)$ needed for the correction step. If implemented by forming $b - Ax$ and multiplying by A^T, this calculation costs an additional $2mnr$ multiplications (one multiplication by A, one by A^T) for every right-hand side. This means that the total work associated with each right-hand side for CSNE is approximately $3mnr$, which is larger by a factor of three than the comparable work for the normal equations, and also larger by a factor of 1.5 than the comparable work for any of the QR-based methods.

Mathematically, the vector needed in the correction step of CSNE can also be written as $A^Tb - A^TAx$, which would seem to be suitable for our application, since the vector A^Tb must be computed for SNE and the matrix A^TA can be calculated efficiently using (3.1) and saved. Unfortunately, Björck's error analysis does *not* apply unless

the residual $b - Ax$ is formed explicitly, so that there appears to be no efficient way to gain the benefits in accuracy of CSNE for our problem.

6. Summary of work and storage for simplified problem. Table 6.1 contains the dominant multiplication counts for all methods (expressed in terms of m, r, n, and d), separated into the work associated with processing A (the column labeled "A") and d right-hand sides. To perform CSNE, R_A can be calculated with struct1, struct2, or struct3; we have arbitrarily shown CSNE with struct3. *The structured normal-equation approach always requires the least work.* The dominance of work involving the right-hand side means that, as the number of right-hand sides increases, the ratio of work required by structured NE, QR-based methods, and CSNE becomes 1:2:3. Eventually, if the number of right-hand sides becomes sufficiently large, even the savings in structured NE are swamped by the work needed to process the right-hand sides.

TABLE 6.1
Work (multiplications) for all methods.

Method	A	d right-hand sides
Unstructured NE	$\frac{1}{2}mrn^2 + \frac{1}{6}n^3$	$dmnr$
Unstructured QR	$mrn^2 - \frac{1}{3}n^3$	$2dmnr$
Structured NE	$\frac{1}{2}(m+r)n^2 + \frac{1}{6}n^3$	$dmnr$
struct1	$mn^2 + \frac{1}{3}rn^3 - \frac{1}{3}n^3$	$2dmnr$
struct2	$(m+r)n^2 - \frac{2}{3}n^3 + \frac{1}{6}n^4$	$d(2mnr + rn^2 - \frac{1}{3}n^3)$
struct3	$(m+r)n^2 - \frac{2}{3}n^3 + \frac{1}{12}n^4$	$d(2mnr + rn^2 - \frac{2}{3}n^3)$
CSNE (with struct3)	$(m+r)n^2 - \frac{2}{3}n^3 + \frac{1}{12}n^4$	$3dmnr$

Table 6.2 shows the work for our typical problem with $m = 500$, $r = 200$, and $n = 60$, for a single right-hand side and 20 right-hand sides. The column labeled "A" contains the work associated with processing A.

TABLE 6.2
Work (multiplications) for typical system identification problem.

Method	A	$d = 1$	$d = 20$	Total, $d = 1$	Total, $d = 20$
Unstructured NE	1.8×10^8	6.0×10^6	1.2×10^8	1.9×10^8	3.0×10^8
Unstructured QR	3.6×10^8	1.2×10^7	2.4×10^8	3.7×10^8	6.0×10^8
Structured NE	1.3×10^6	6.0×10^6	1.2×10^8	7.3×10^6	1.2×10^8
struct1	1.6×10^7	1.2×10^7	2.4×10^8	2.8×10^7	2.6×10^8
struct2	4.6×10^6	1.3×10^7	2.5×10^8	1.7×10^7	2.6×10^8
struct3	3.5×10^6	1.3×10^7	2.5×10^8	1.6×10^7	2.5×10^8
CSNE (with struct3)	3.5×10^6	1.8×10^7	3.6×10^8	2.2×10^7	3.6×10^8

Table 6.3 gives the dominant storage requirements for each structured method; it also shows the specific storage needed when solving a typical problem with $m = 500$, $r = 200$, and $n = 60$, for one right-hand side and 20 right-hand sides. The storage

for version 1 of the three structured QR methods is unaffected by the number of right-hand sides, since only one right-hand side is processed at a time.

TABLE 6.3
Storage required by structured methods.

Method	A	d right-hand sides	$d = 1$	$d = 20$
Structured NE	$mn + nr$	mr	5.0×10^5	5.0×10^5
struct1 (version 1)	$mn + nr + \frac{1}{2}rn^2$	mr	5.0×10^5	5.0×10^5
struct1 (version 2)	$mn + nr$	$mr + dnr$	1.5×10^5	3.8×10^5
struct2 (version 1)	$mn + nr + \frac{1}{3}n^3$	mr	2.1×10^5	2.1×10^5
struct2 (version 2)	$mn + nr$	$mr + dn^2$	1.5×10^5	2.1×10^5
struct3 (version 1)	$mn + nr + \frac{1}{6}n^3$	mr	1.8×10^5	1.8×10^5
struct3 (version 2)	$mn + nr$	$mr + \frac{1}{2}dn^2$	1.4×10^5	1.8×10^5
CSNE (with struct3)	$mn + nr + \frac{1}{2}n^2$	mr	1.4×10^5	1.4×10^5

7. Numerical results. For completeness, we briefly describe selected numerical results that confirm the expected properties of the methods. Our numerical tests were run on a Silicon Graphics 4D/440VGX with 40 MHz MIPS R3000 cpu chips and binary IEEE arithmetic, on a single processor. The calculations were performed in double precision, corresponding to approximately 16 decimal digits, and all methods described were implemented in Fortran 77. Some care is needed in storing and representing the special sequences of Householder transformations for the structured QR methods, but for the most part implementation is relatively straightforward.

7.1. Time. The estimates of work given in §6 provide reasonably good predictions of the observed relative times. Consider solving a randomly generated problem in which $m = 120$, $r = 80$, $n = 50$, and $d = 1$; we chose these dimensions because they are large enough to be interesting and are consistent with our initial assumption that $m > r > n$, yet are small enough to allow us to make a very large number of numerical experiments. Furthermore, as we shall see, the numerical testing produced no surprises.

Table 7.1 shows the multiplication counts from the formulas in Table 6.1 associated with the various structured methods for this problem.

TABLE 7.1
Work estimates (multiplications) for random problem.

Method	Processing A	$d = 1$	Total for $d = 1$
Normal equations	2.7×10^5	4.8×10^5	7.5×10^5
struct1	3.7×10^6	9.6×10^5	4.6×10^6
struct2	1.5×10^6	1.1×10^6	2.6×10^6
struct3	9.5×10^5	1.1×10^6	2.0×10^6
CSNE (with struct3)	9.5×10^5	1.4×10^6	2.4×10^6

The actual running times (in seconds) for these methods, averaged over approximately fifty sets of experiments, are given in Table 7.2. They are broadly consistent

with the predictions of Table 6.1, except that the times needed to process the right-hand side in the QR-based methods are smaller than expected. This appears to be attributable to compiler optimization of loops in calculations involving the right-hand side.

TABLE 7.2
Actual times (seconds) to solve random problem.

Work	Processing A	$d = 1$	Total for $d = 1$
Normal equations	0.27	0.63	0.90
struct1	5.3	0.82	6.1
struct2	1.8	0.88	2.7
struct3	1.1	0.83	1.9
CSNE	1.1	1.4	2.5

7.2. Accuracy. As expected, for ill-conditioned problems the structured QR methods and CSNE consistently produced more accurate answers than the normal equations. We illustrate their behavior with a compatible ill-conditioned problem for which $\text{cond}(A^TA) \approx 10^{13}$, $n = 5$, and the exact solution is

$$x = (5.25,\ -5.25,\ 5.25,\ -5.25,\ 5.25)^T.$$

The (noticeably inaccurate) solution computed by the normal equations is

$$x_{NE} = (\ 5.2769095995, -5.2376267727, 5.271286724, -5.2308866252, 5.2168334259\ )^T.$$

The much better solution produced by struct2 is

$$x_{QR} = (\ 5.249999998, -5.250000001, 5.249999998, -5.250000002, 5.250000003\ )^T.$$

(All the solutions computed by QR-based methods are of comparable accuracy.)

The solution produced by the seminormal equations with the matrix R_A from struct2 is

$$x_s = (\ 5.2505414469, -5.2497510388, 5.2504283093, -5.2496119765, 5.2493293168\ )^T$$

and is indeed more accurate than x_{NE}. As predicted by the analysis in [4], however, it does not approach the accuracy of x_{QR}. After the correction step, the greatly improved solution, now comparable in accuracy to x_{QR}, is

$$x_c = (\ 5.2499999995, -5.2500000002, 5.2499999996, -5.2500000004, 5.2500000006\ )^T.$$

In some of our experiments, CSNE produced a good solution even when the normal equations failed completely because of a breakdown in the Cholesky factorization of the computed A^TA.

8. The complex problem. In this section, we deal briefly with the full complex problem, repeated for convenience:

$$\text{(8.1)} \qquad \text{minimize} \quad \sum_{i=1}^{m}\sum_{j=1}^{r}\left|\left(\sum_{k=1}^{n} G_{ik}H_{jk}z_k + J_{ik}H_{jk}^{*}z_k^{*}\right) - b_{ij}\right|^2.$$

It is important to notice immediately that the problem statement (8.1) does not fit the standard formulation of a least-squares problem in terms of a matrix and a vector. (The tricky part is the appearance in the second term of complex conjugate elements of H and z.)

8.1. A formulation with real variables. Since serious numerical computation in IEEE arithmetic is typically performed in double precision, but relatively few compilers support the type "double precision complex," a formulation in real arithmetic is desirable in many contexts. A QR-based solution approach involving some complex arithmetic is described in §8.3.

To express (8.1) in a standard form, we order z as $x_1, y_1, \ldots, x_n, y_n$, and then write all quantities in terms of their real and imaginary parts. In particular, let

$$z = x + iy, \quad G = C + iD, \quad J = E + iF, \quad \text{and} \quad H = K + iL, \tag{8.2}$$

where all vectors and matrices on the right of each equation are real. The all-real formulation is obtained by observing patterns in the coefficients: for example, x_1 and y_1 appear only in the terms $g_{11}h_{11}z_1 + j_{11}h_{11}^*z_1^*$ of (8.1), which can be expanded as follows:

$$(c_{11} + id_{11})(k_{11} + i\ell_{11})(x_1 + iy_1) \quad + \quad (e_{11} + if_{11})(k_{11} - i\ell_{11})(x_1 - iy_1).$$

Multiplying out this expression and splitting into real and imaginary parts, we see that the coefficients in the real part are

$$\begin{aligned} (c_{11} + e_{11})k_{11} + (-d_{11} + f_{11})\ell_{11} \qquad & \text{for } x_1 \\ -(c_{11} + e_{11})\ell_{11} + (-d_{11} + f_{11})k_{11} \qquad & \text{for } y_1. \end{aligned}$$

In the imaginary part, these coefficients are

$$\begin{aligned} (d_{11} + f_{11})k_{11} + (c_{11} - e_{11})\ell_{11} \qquad & \text{for } x_1 \\ -(d_{11} + f_{11})\ell_{11} + (c_{11} - e_{11})k_{11} \qquad & \text{for } y_1. \end{aligned}$$

Similar relations apply for all the variables, which implies that the original complex problem can be formulated in terms of two real matrices P_1 and P_2, both $m \times 2n$. For a generic matrix M, the notation "$(M)_j$" means column j of M. The matrix P_1 is defined by interleaving the columns of the two $n \times n$ real matrices $(C + E)$ and $(-D + F)$; P_2 includes the interleaved columns of $(D + F)$ and $(C - E)$:

$$P_1 = \begin{pmatrix} (C+E)_1 & (-D+F)_1 & \cdots & (C+E)_n & (-D+F)_n \end{pmatrix}, \tag{8.3a}$$

$$P_2 = \begin{pmatrix} (D+F)_1 & (C-E)_1 & \cdots & (D+F)_n & (C-E)_n \end{pmatrix} \tag{8.3b}$$

Let $\bar{P}$ denote the $2m \times 2n$ matrix

$$\bar{P} = \begin{pmatrix} P_1 \\ P_2 \end{pmatrix}, \tag{8.4}$$

and let $\bar{H}^{(i)}$, $i = 1, \ldots, r$, be the $2n \times 2n$ matrix that is zero except for 2×2 blocks along the diagonal, satisfying

$$j\text{th block of } \bar{H}^{(i)} = \begin{pmatrix} k_{ij} & -\ell_{ij} \\ \ell_{ij} & k_{ij} \end{pmatrix}. \tag{8.5}$$

An entirely real least-squares matrix A of dimension $2mr \times 2n$ is then given by

$$A = \begin{pmatrix} \bar{P}\bar{H}^{(1)} \\ \vdots \\ \bar{P}\bar{H}^{(r)} \end{pmatrix}. \tag{8.6}$$

The least-squares matrix in (8.6) has twice as many rows and twice as many columns (i.e., four times as many real elements) as the matrix in the simplified real problem (2.4). An additional difference is that the matrices $\bar{H}^{(i)}$ in (8.6) are block-diagonal rather than diagonal.

8.2. The normal equations revisited. For the simplified real problem, the normal-equation matrix can be formed from the elements of G^TG and H^TH; see (3.1) and (3.2). With A of (8.6), however, expressing A^TA in terms of $\bar{P}$, K, and L is less straightforward. Writing out A^TA, we have

$$A^TA = \bar{H}^{(1)T}\bar{P}^T\bar{P}\bar{H}^{(1)} + \cdots + \bar{H}^{(r)T}\bar{P}^T\bar{P}\bar{H}^{(r)}. \tag{8.7}$$

Using the special structure of $\bar{H}^{(i)}$ (see (8.5)), it can be shown that the (i,j) element of A^TA is a linear combination of four elements of $\bar{P}^T\bar{P}$; the coefficients in the linear combination involve elements of the three matrices K^TK, K^TL, and L^TL, where L and K are defined in (8.2). Let

$$\Sigma^{(1)} = K^TK, \quad \Sigma^{(2)} = K^TL, \quad \Sigma^{(3)} = L^TK, \quad \text{and} \quad \Sigma^{(4)} = L^TL. \tag{8.8}$$

The (i,j) element of $\Sigma^{(k)}$ will be denoted by $\sigma_{ij}^{(k)}$.

The easiest way to characterize the structure of A^TA is to think of it and $\bar{P}^T\bar{P}$ as divided into n^2 distinct 2×2 "boxes." (In general, the boxes are not symmetric, except for those along the diagonal.) Suppose that

$$\Phi = \text{box}_{ij} \text{ of } \bar{P}^T\bar{P} = \begin{pmatrix} \phi_{ij}^{(1)} & \phi_{ij}^{(2)} \\ \phi_{ij}^{(3)} & \phi_{ij}^{(4)} \end{pmatrix} \text{ and } \text{box}_{ij} \text{ of } A^TA = \begin{pmatrix} \alpha_{ij}^{(1)} & \alpha_{ij}^{(2)} \\ \alpha_{ij}^{(3)} & \alpha_{ij}^{(4)} \end{pmatrix}.$$

Then $\alpha_{ij}^{(1)}$ is given by

$$\alpha_{ij}^{(1)} = \sigma_{ij}^{(1)}\phi_{ij}^{(1)} + \sigma_{ij}^{(2)}\phi_{ij}^{(2)} + \sigma_{ij}^{(3)}\phi_{ij}^{(3)} + \sigma_{ij}^{(4)}\phi_{ij}^{(4)}. \tag{8.9}$$

Formulas similar to (8.9) can also be derived that express $\alpha_{ij}^{(2)}$, $\alpha_{ij}^{(3)}$, and $\alpha_{ij}^{(4)}$ in terms of the (i,j) elements of the Σ matrices of (8.8) and the four elements of Φ, so that the normal-equation matrix can be formed using the structure of A almost as efficiently as in the simplified real case (see Table 8.1).

8.3. A QR-based approach using some complex arithmetic. If complex arithmetic in an appropriate precision is available, a QR-based method can be derived by expressing the problem (8.1) in a form where the least-squares matrix is partitioned into blocks, each containing the same complex matrix on the left.

Suppose that the columns of G and J are interleaved to form a single complex matrix $\hat{G}$ with m rows and $2n$ columns:

$$\hat{G} = \begin{pmatrix} g_1 & j_1 & g_2 & j_2 & \cdots & g_n & j_n \end{pmatrix}, \tag{8.10}$$

where g_s and j_s denote the sth columns of G and J. Then problem (8.1) can be written as minimizing the two-norm of a complex vector with r blocks, where the sth block is

$$\hat{G}\begin{pmatrix} h_{s1}z_1 \\ h_{s1}^*z_1^* \\ \vdots \\ h_{sn}z_n \\ h_{sn}^*z_n^* \end{pmatrix} - b^{(s)}. \tag{8.11}$$

As in method struct1 of Section 4.1, the complex QR factorization of $\hat{G}$ can be used to create a matrix with many fewer nonzeros. Let $\hat{Q}$ be the associated $m \times m$ unitary matrix $\hat{Q}$, and $\hat{R}$ the square upper-triangular complex matrix of dimension $2n$ (assuming that $m > 2n$). This calculation requires $4mn^2 - \frac{8}{3}n^3$ complex multiplications.

If the block-diagonal matrix $\text{diag}(\hat{Q}, \ldots, \hat{Q})$ is applied to the complete least-squares matrix corresponding to (8.11) and zero rows are removed, the sth block of the complex vector whose norm is to be minimized becomes

$$\hat{R} \begin{pmatrix} h_{s1} z_1 \\ h_{s1}^* z_1^* \\ \vdots \\ h_{sn} z_n \\ h_{sn}^* z_n^* \end{pmatrix} - \tilde{b}^{(s)}, \tag{8.12}$$

where $\tilde{b}^{(s)}$ is a transformed and condensed version of $b^{(s)}$.

We now write all matrices and vectors in terms of their real and imaginary parts. Let $z = x + iy$, $H = K + iL$, and $\hat{R} = T + iU$, where T and U are upper-triangular. The portion of (8.12) involving the unknowns then looks like

$$\begin{pmatrix} t_{11} + iu_{11} & t_{12} + iu_{12} & \cdots & t_{1,2n} + iu_{1,2n} \\ & t_{22} + iu_{22} & \cdots & t_{2,2n} + iu_{2,2n} \\ & & \ddots & \vdots \\ & & & t_{2n,2n} + iu_{2n,2n} \end{pmatrix} \begin{pmatrix} (k_{s1} + \ell_{s1})(x_1 + iy_1) \\ (k_{s1} - \ell_{s1})(x_1 - iy_1) \\ \vdots \\ (k_{sn} + \ell_{sn})(x_n + iy_n) \\ (k_{sn} - \ell_{sn})(x_n - iy_n) \end{pmatrix}.$$

If we split each row of this matrix-vector product into its real and imaginary parts and order the components of the unknown vector as

$$(x_1, y_1, x_2, y_2, \ldots, x_n, y_n)^T,$$

the problem can be expressed in terms of an entirely real least-squares matrix with r blocks, each with $4n$ rows and $2n$ columns. If the rows of each block are ordered so that x_1 and y_1 occur only in the leading rows, then the first four rows of each block contain nonzero coefficients multiplying x_1 and y_1. Because T and U are triangular, rows 5 through 8 contain zeros in the first two columns, and so on. For $n = 3$, the structure of a typical block is displayed in the first form shown in Fig. 8.1. Alternatively, if all rows corresponding to the real parts are placed first in each block, followed by all rows corresponding to the imaginary parts, a typical block has the structure shown in the second form in Fig. 8.1.

As in procedure struct1 from Section 4.1, a special sequence of Householder transformations can be constructed to reduce the least-squares matrix composed of blocks like those in Fig. 8.1 to upper-triangular form, at a cost of $\frac{16}{3}rn^3$ multiplications.

8.4. QR-based methods using only real arithmetic. Although a QR-based algorithm could be defined by performing a QR decomposition of the matrix

$$\Psi = (\ c_1\ d_1\ e_1\ f_1\ c_2\ d_2\ e_2\ f_2\ \ldots\ c_n\ d_n\ e_n\ f_n\ ), \tag{8.13}$$

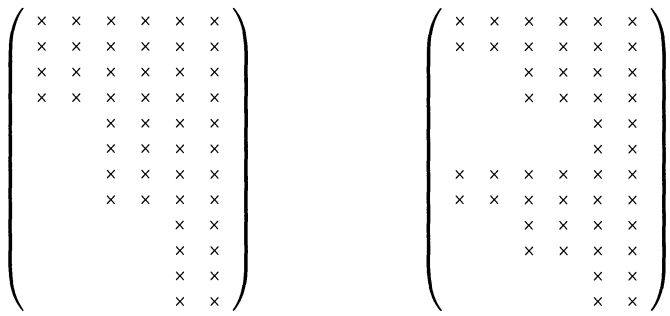


FIG. 8.1. *Blocks of partially reduced matrix for the complex problem.*

less work and storage are required if we use instead the QR factorizations of P_1 and P_2, the two $m \times 2n$ matrices defined in (8.3):

$$Z_1 P_1 = \tilde{S}_1 = \begin{pmatrix} S_1 \\ 0 \end{pmatrix}, \quad Z_2 P_2 = \tilde{S}_2 = \begin{pmatrix} S_2 \\ 0 \end{pmatrix},$$

where S_1 and S_2 are $2n \times 2n$ upper triangles. (We assume that $m > 2n$.) Let $\bar{Z}$ and $\tilde{Z}$ be defined as

$$\bar{Z} = \begin{pmatrix} Z_1 & 0 \\ 0 & Z_2 \end{pmatrix} \quad \text{and} \quad \tilde{Z} = \operatorname{diag}(\bar{Z}, \ldots, \bar{Z}).$$

Then, multiplying A of (8.6) on the left by $\tilde{Z}$ and removing zero rows, the result is a partially reduced matrix (denoted by $\tilde{A}$) with r blocks, each with $4n$ rows:

$$\tilde{A} = \begin{pmatrix} \bar{S}\bar{H}^{(1)} \\ \vdots \\ \bar{S}\bar{H}^{(r)} \end{pmatrix}, \quad \text{where } \bar{S} \text{ is defined as } \quad \bar{S} = \begin{pmatrix} S_1 \\ S_2 \end{pmatrix}. \tag{8.14}$$

It follows from the double-triangular structure of $\bar{S}$ in (8.14) and the block-diagonal structure (8.5) of $\bar{H}^{(i)}$ that a typical $4n$-row block of $\bar{S}\bar{H}^{(i)}$ has the second structure shown in Fig. 8.1 for $n = 3$. A method analogous to struct1 can then be applied to produce a final upper-triangular form. Note that, if we had worked with Ψ of (8.13), each block would have twice as many rows, $4n$ each for the real and imaginary parts of the equations. Table 8.1 compares the operation counts (in real multiplications) for the structured normal-equation method of §8.2 and the analogue of struct1 just described.

TABLE 8.1
Work (in real multiplications) for complex problem.

Method	A	d right-hand sides
Structured NE	$(4m + 2r + 8)n^2 + \frac{4}{3}n^3$	$4dmnr$
Struct1 analogue (complex case)	$8mn^2 + \frac{16}{3}rn^3 - \frac{16}{3}n^3$	$8dmnr$

Alternatively, we may wish to proceed along the lines of struct2 and struct3, and to develop a method in which a factorization related to H plays a role. Recall that

$H = K + iL$, and let $\tilde{H}$ denote the $r \times 2n$ matrix in which the real and imaginary parts of the columns of H are interleaved:

$$\tilde{H} = \begin{pmatrix} k_1 & \ell_1 & \cdots & k_n & \ell_n \end{pmatrix}. \tag{8.15}$$

Rearranging $\tilde{A}$ of (8.14) into a convenient form with $\tilde{H}$ on the left in each block is more complicated than for the simplified real problem for two reasons: the matrices $\bar{H}^{(k)}$ in (8.14) are block-diagonal rather than diagonal; and $\bar{S}$ contains two triangular matrices, S_1 and S_2.

For a generic upper-triangular matrix S of dimension $2n \times 2n$, let $\hat{S}^{(i)}$ denote a special $2n \times 2n$ block-diagonal matrix with 2×2 blocks containing elements from the ith row of S:

$$\hat{S}^{(i)} = \begin{pmatrix} s_{i1} & s_{i2} & & & \\ s_{i2} & -s_{i1} & & & \\ & & s_{i3} & s_{i4} & \\ & & s_{i4} & -s_{i3} & \\ & & & & \ddots \end{pmatrix}. \tag{8.16}$$

The triangularity of S means that one or both elements of some of the blocks of $\hat{S}^{(i)}$ will be zero, depending on the value of i. For example, when $n = 2$, we have

$$S = \begin{pmatrix} s_{11} & s_{12} & s_{13} & s_{14} \\ & s_{22} & s_{23} & s_{24} \\ & & s_{33} & s_{34} \\ & & & s_{44} \end{pmatrix}, \quad \hat{S}^{(1)} = \begin{pmatrix} s_{11} & s_{12} & & \\ s_{12} & -s_{11} & & \\ & & s_{13} & s_{14} \\ & & s_{14} & -s_{13} \end{pmatrix},$$

$$\hat{S}^{(2)} = \begin{pmatrix} 0 & s_{22} & & \\ s_{22} & 0 & & \\ & & s_{23} & s_{24} \\ & & s_{24} & -s_{23} \end{pmatrix}, \quad \hat{S}^{(3)} = \begin{pmatrix} 0 & 0 & & \\ 0 & 0 & & \\ & & s_{33} & s_{34} \\ & & s_{34} & -s_{33} \end{pmatrix}.$$

It follows from the block-diagonal structure of $\bar{H}^{(k)}$ (see (8.5)) and the stacked-triangular structure of $\bar{S}$ (see (8.14)) that rows 1, $2n + 1$, $4n + 1$, and so on, of the matrix $\tilde{A}$ in (8.14) can be grouped together into the single block $\tilde{H}\hat{S}_1^{(1)}$. Similarly, we may group together rows $n + 1$, $3n + 1$, $5n + 1$, and so on, producing a block of the form $\tilde{H}\hat{S}_2^{(1)}$. In this way, we create a row-permuted version of $\tilde{A}$, called $\bar{A}$, in which the matrix $\tilde{H}$ of (8.15) appears on the left in each block:

$$\bar{A} = \begin{pmatrix} \tilde{H}\hat{S}_1^{(1)} \\ \tilde{H}\hat{S}_2^{(1)} \\ \vdots \\ \tilde{H}\hat{S}_1^{(n)} \\ \tilde{H}\hat{S}_2^{(n)} \end{pmatrix}. \tag{8.17}$$

By analogy with struct2 and struct3, we could use a QR or QL factorization of $\tilde{H}$ to reduce $\bar{A}$ further.

Just as in the simplified real problem, the matrices $\hat{S}^{(k)}$ include more zero elements as we move down the rows of $\bar{A}$. However, in this case two zero columns are added at the front after every two blocks, rather than one new zero column after every block, as in Figs. 4.2 and 4.3.

9. Issues related to parallel computation. Our discussion so far has indicated that many of the computations associated with the special structure discussed in this paper involve *standard dense techniques.* Because so much research has been devoted to developing ultra-efficient machine-specific dense linear algebraic procedures, our special structure can thus be viewed as generically well suited to parallel and high-performance architectures. In particular, the dense Cholesky and orthogonal-triangular factorizations can be obtained using appropriately tailored Level-3 BLAS (basic linear algebra subroutines); see, for example, [1]. An excellent and comprehensive discussion of parallel linear algebraic computation, both dense and sparse, is given in [6].

Beyond effective use of specialized BLAS, the most promising place to exploit a particular architecture is in handling the right-hand side—the dominant cost of the calculation. Before discussing specific methods, we state the obvious fact that the d right-hand sides can always be processed independently; this feature immediately implies a strategy for a distributed environment.

9.1. Normal and corrected seminormal equations. In forming $A^T b$ for the normal equations and CSNE in the real case, each right-hand side vector can be grouped into r separate row blocks of dimension m. Since $r = 200$ for our typical problem, this calculation provides an opportunity to gain speed on massively parallel machines. With d right-hand sides, the matrix G^T can be applied simultaneously to d columns of B within each of the r blocks. Calculation of the two matrices $G^T G$ and $H^T H$ needed to obtain $A^T A$ using (3.2) can, if possible, be performed simultaneously.

For the complex problem, the normal-equation approach described in §8.2 requires the four matrices $\bar{P}^T \bar{P}$, $K^T K$, $L^T L$, and $K^T L$, which may be calculated simultaneously; see (8.4) and (8.8). In processing the right-hand side, $\bar{P}^T$ may be applied to the $2r$ blocks of each vector b simultaneously and to d columns of B within each block.

9.2. Structured QR methods. With all of the structured QR methods for the simplified real problem, we need to calculate an r-block vector $\tilde{b}$ whose ith block is derived from $Qb^{(i)}$. As in forming $A^T b$ for the normal equations, $\tilde{b}$ can be calculated by applying the same matrix Q to d columns at once in each of the r blocks. Thus Level-3 BLAS may be used on r processors in a multiprocessing, MIMD (multiple-instruction multiple-data) environment. Even when $d = 1$, Q can be applied to the r right-hand side blocks simultaneously.

With struct2 and struct3, there is further opportunity for parallelism in treating the right-hand side. In these methods, each block of a rearranged $\tilde{b}$ needs to be transformed by either Y of (4.8) or V of (4.11). If we view each right-hand side b as a matrix, say Ω, with m rows and r columns, stored by columns, then the calculation of $\tilde{b}$ involves first applying Q to the r columns of Ω simultaneously, producing the matrix $\tilde{\Omega}$. Because the permutation P_R in (4.7) groups every nth row together, application of Y or V to the rearranged version of $\tilde{b}$ is equivalent to applying Y or V from the right to the first n rows of $\tilde{\Omega}$. Moreover, each of the rows can be processed independently.

Within struct2 and struct3, the two required orthogonal-triangular factorizations of G and H can be performed simultaneously to take advantage of a two-processor machine. The techniques in §8.4 require either two or three QR factorizations (of P_1,

P_2, and $\tilde{H}$); these can be calculated independently, each using Level-3 BLAS.

Depending on the architecture and the number of processors, the special sequences of Householder transformations in the three structured QR techniques can be applied to d right-hand sides simultaneously. Since a much higher proportion of the work associated with struct2 and struct3 involves calculations that are obviously parallelizable, these methods would probably be preferred in a parallel environment.

Parallelism can also be exploited in performing the final reduction to triangular form for the structured QR methods. In struct1, for example, $\tilde{A}$ of (4.4) could be split into $\lfloor r/2 \rfloor$ pairs of blocks, each containing two upper-triangular matrices. Within each pair, the "bottom" upper-triangular matrix could simultaneously be eliminated using the top matrix. The remaining matrices could then similarly be grouped by pairs in blocks, and so on.

For struct2 and struct3, analogous divide-and-conquer strategies can be developed by overlapping transformations that annihilate disjoint portions of the partially reduced matrices. Only n Householder transformations are needed to reduce the matrix $\hat{A}$ of Fig. 4.2 within struct2, whereas $\frac{1}{2}n^2$ separate transformations are applied in reducing $\breve{A}$ of Fig. 4.3 within struct3. Hence, although struct3 requires fewer scalar multiplications, it appears to be somewhat less well suited to a parallel environment.

It turns out, however, that transformations in the last stage of struct3 can be overlapped to the extent that only $2n-2$ consecutive sets of simultaneous transformations are needed. We illustrate the overlapping strategy with the 10 frames (partially reduced matrices) shown in Fig. 4.4, where the triangles are numbered starting at the top. Annihilation of the $(10,3)$ element using triangle 3 (shown in frame 3) can be overlapped with the annihilation of the $(9,4)$ element using triangle 2 (shown in frame 4); next, elimination of elements in column 3 of triangle 3 (frame 5) can be overlapped with annihilation of the $(7,4)$ element (frame 7); and then column 2 in triangle 3 (frame 6) can be annihilated simultaneously with column 3 in triangle 2 (frame 8). In general, with n shrinking triangles in $\breve{A}$, at step j with $j < n$ we can simultaneously eliminate column n of triangle $n-j+1$; column $n-1$ of triangle $n-j+2$; column $n-2$ of triangle $n-j+3$; and so on. At step j for $j \geq n$, we can simultaneously eliminate column $2n-j-1$ of triangle 2, column $2n-j-2$ of triangle 3, column $2n-j-3$ of triangle 4, and so on.

10. Separable nonlinear least-squares. In this last section, we turn to an apparently different problem: separable nonlinear least-squares problems with multiple data sets. We assume that there are ℓ linear parameters c and n nonlinear parameters x, and that the model function to be fitted to the ith observation is

$$\eta_i(c,x) = \sum_{j=1}^{\ell} c_j \phi_{ij}(x). \tag{10.1}$$

The nonlinear function $\phi_{ij}(x)$ is called the jth *term* in the model; if ϕ_{ij} is independent of x, the jth term is purely linear.

10.1. Variable projection methods for separable least-squares. Fitting the model (10.1) to m data values $\{b_i\}$ in the two-norm is a nonlinear least-squares problem with $\ell + n$ variables:

$$\underset{c\in\mathcal{R}^{\ell},x\in\mathcal{R}^{n}}{\text{minimize}} \sum_{i=1}^{m}\Big(\sum_{j=1}^{\ell} c_j\phi_{ij}(x) - b_i\Big)^2 = \underset{c,x}{\text{minimize}} \|\Phi(x)c - b\|_2^2, \tag{10.2}$$

where $\Phi(x)$ is a matrix whose (i,j) element is $\phi_{ij}(x)$.

When a large proportion of the variables are linear, the "variable projection" method proposed by Golub and Pereyra [8] exploits separability by splitting the problem into two subproblems—a nonlinear least-squares problem with n variables, and a linear least-squares problem with ℓ variables. If the nonlinear variables $\bar{x}$ are fixed, the linear parameter c that solves the linear least-squares problem of minimizing $\|\Phi(\bar{x})c - b\|_2^2$ is given by $c = \Phi^+(\bar{x})b$, where Φ^+ denotes the pseudoinverse of Φ. Substituting this expression in (10.2), the optimal nonlinear parameter vector x solves the nonlinear least-squares problem

$$\underset{x\in\mathcal{R}^n}{\text{minimize}} \quad \| (\Phi(x)\Phi^+(x) - I)b \|_2^2. \tag{10.3}$$

An equivalent formulation of (10.3), proposed by Kaufman [10], is

$$\underset{x\in\mathcal{R}^n}{\text{minimize}} \quad \| Z(x)^T b \|_2^2, \tag{10.4}$$

where $Z(x)$ denotes an orthonormal basis for the null space of $\Phi(x)$.

The nonlinear functions in (10.3) and (10.4), as well as the associated Jacobian matrices, can be obtained from the QR factorization of the $m \times \ell$ matrix Φ; see [10] for details. After the optimal vector $\bar{x}$ is obtained from (10.3) or (10.4), the optimal vector of linear parameters then solves the ℓ-variable linear least-squares problem of minimizing $\|\Phi(\bar{x})c - b\|_2^2$.

10.2. Multiple data sets. As indicated in §1.2, we are interested in separable problems with multiple sets of observations and linear parameters, such that the nonlinear parameters are common to all data sets. Let m denote the number of observations per data set and r the number of data sets. The problem is

$$\underset{x\in\mathcal{R}^n}{\text{minimize}} \sum_{k=1}^{r} \| \Phi(x)c^{(k)} - b^{(k)} \|_2^2, \tag{10.5}$$

where Φ is the matrix from (10.2) and $c^{(k)}$ is the ℓ-vector of linear parameters for the kth data set. Golub and LeVeque [7] noted that this problem may be expressed as minimizing the two-norm of the mr-dimensional vector

$$\begin{pmatrix} \Phi(x) & & \\ & \ddots & \\ & & \Phi(x) \end{pmatrix} \begin{pmatrix} c^{(1)} \\ \vdots \\ c^{(r)} \end{pmatrix} - \begin{pmatrix} b^{(1)} \\ \vdots \\ b^{(r)} \end{pmatrix}. \tag{10.6}$$

Since $\Phi(x)$ is the same for all data sets, it is easy to see that the QR factorization of the matrix in (10.6) can be obtained directly from the QR factorization of Φ.

If a variable projection method based on (10.4) is applied to the nonlinear least-squares problem (10.6) associated with multiple data sets, the Jacobian matrix is $mr \times n$, and can be divided into r blocks, each $m \times n$:

$$J = \begin{pmatrix} J^{(1)} \\ \vdots \\ J^{(r)} \end{pmatrix}. \tag{10.7}$$

It is shown in [11] that the kth block $J^{(k)}$ has the form

$$J^{(k)} = \begin{pmatrix} \Gamma_1 c^{(k)} & \Gamma_2 c^{(k)} & \cdots & \Gamma_n c^{(k)} \end{pmatrix}, \tag{10.8}$$

where Γ_s, $s = 1, \ldots, n$ is an $m \times \ell$ matrix and $c^{(k)}$ is the current estimate of the linear parameters for the kth data set. The jth column of Γ_s is a transformed version of the m-vector of partial derivatives of ϕ_{ij}, $i = 1, \ldots, m$, with respect to x_s. Note that the same matrices $\{\Gamma_s\}$ occur in all blocks of the Jacobian.

Many methods for nonlinear least-squares calculate the next iterate by solving a linear least-squares subproblem involving the Jacobian matrix. Hence our discussion of structured linear least-squares problems will be directly relevant if the Jacobian has a suitable structure. In the next two subsections, we explore application of the techniques presented in §§3–5 to the Jacobian (10.7).

10.3. Models with a special structure. Many models of practical interest display the following property:

(10.9) *Each nonlinear variable appears in exactly one term of the model.*

This problem class is illustrated by the following two models:

$$\eta_i = c_1 \exp(x_1 t_i) + c_2 \exp(x_2 t_i); \tag{10.10}$$

$$\eta_i = c_1 \exp(-x_1 t_i) + c_2 \exp(-x_2 (t_i - x_3)^2) + c_3 \exp(-x_4 (t_i - x_5)^2). \tag{10.11}$$

Some models satisfying (10.9) have the further property that each term includes exactly one nonlinear variable. Since the linear and nonlinear parameters "match" in this case—in particular, n must be equal to ℓ—we refer to these as "matched" models. Model (10.10) is a matched model. (We assume without loss of generality that the kth nonlinear parameter in a matched model appears in the kth term.)

When (10.9) holds, each matrix Γ_s in (10.8) contains only one nonzero column. Let g_s denote the single nonzero column in Γ_s, and define the $m \times n$ matrix G as

$$G = \begin{pmatrix} g_1 & g_2 & \cdots & g_n \end{pmatrix}. \tag{10.12}$$

To express the Jacobian (10.7) in terms of G, we define $\hat{c}^{(k)}$ as the n-vector whose sth component is the element of $c^{(k)}$ corresponding to the term in which x_s appears. In a matched model, $\hat{c}^{(k)} = c^{(k)}$. Otherwise, however, elements of $c^{(k)}$ are *repeated* in $\hat{c}^{(k)}$. With the model (10.11), for example,

$$(\hat{c}^{(k)})^T = \begin{pmatrix} c_1^{(k)} & c_2^{(k)} & c_2^{(k)} & c_3^{(k)} & c_3^{(k)} \end{pmatrix}. \tag{10.13}$$

Let $\hat{C}$ denote the $r \times n$ matrix whose kth row is the transpose of $\hat{c}^{(k)}$, and let $\hat{C}^{(k)}$ denote the diagonal matrix whose elements are those of $\hat{c}^{(k)}$. Then the Jacobian of (10.7) can be expressed as a stack of r blocks, each $m \times n$:

$$J = \begin{pmatrix} G\hat{C}^{(1)} \\ \vdots \\ G\hat{C}^{(r)} \end{pmatrix}. \tag{10.14}$$

This matrix has a form similar to (2.4), with $\hat{C}$ playing the role of H.

The structured normal-equation techniques of §3 can obviously be applied directly to a linear least-squares subproblem involving J of (10.14). We now describe how to adapt the QR-based techniques of §4 to the matrix (10.14).

As with the form (2.4), the QR factorization of G can be applied to each block of the Jacobian. After omitting zero rows, the transformed Jacobian has the form

$$\begin{pmatrix} Q_G & & \\ & \ddots & \\ & & Q_G \end{pmatrix} J = \begin{pmatrix} R_G \hat{C}^{(1)} \\ \vdots \\ R_G \hat{C}^{(r)} \end{pmatrix}, \tag{10.15}$$

where Q_G is the orthogonal matrix and R_G is the $n \times n$ upper-triangular matrix from the QR factorization of G. At this stage, the QR factorization of (10.15) can be calculated as in struct1.

Alternatively, if we proceed along the lines of struct2 and struct3, every n rows of (10.15) can be grouped together, as in (4.7). The transformed and row-permuted Jacobian has n blocks, each $r \times n$:

$$\begin{pmatrix} \hat{C} R_G^{(1)} \\ \vdots \\ \hat{C} R_G^{(n)} \end{pmatrix}, \tag{10.16}$$

where $R_G^{(i)}$ is the diagonal matrix whose elements are those of the ith row of R_G.

Although this form is strongly suggestive of (4.7), since $\hat{C}$ appears on the left in every block, some care is needed because $\hat{C}$ will have repeated columns for a nonmatched model, and hence will not have full rank; see (10.13). Consequently, we work instead with the $r \times \ell$ matrix C whose kth row is the transpose of $c^{(k)}$. (In a matched model, $\hat{C} = C$.) Exactly as in §4.2, orthogonal-triangular factors of C can be used in further processing of the Jacobian, as we now illustrate.

Following the methodology of struct3, suppose that C is reduced to a lower triangle. Let Q_C and L_C denote the QL factors of C, where L_C is an $\ell \times \ell$ lower triangle. (We assume that $r \geq \ell$.) Now consider the effect of multiplying each block of the matrix in (10.16) by Q_C. For a matched model, L_C itself will appear on the left in each transformed block. Otherwise, the matrix on the left contains a "stretched" $\ell \times n$ version of L_C with some repeated columns.

Since R_G is upper-triangular, the diagonal matrices $\{R_G^{(k)}\}$ appearing on the right in (10.16) contain one additional zero column at the front as we move down the blocks. After transformation by Q_G and Q_C, the nonzero rows of the Jacobian for a matched model thus have exactly the form of $\breve{A}$ of Fig. 4.3. For nonmatched models satisfying (10.9), the transformed matrix is a stack of lower triangles with some missing rows. For example, with the model (10.11), application of Q_G and Q_C to the Jacobian produces the following structure:

$$\begin{pmatrix} \times & & & & & \\ \times & \times & \times & & & \\ \times & \times & \times & \times & \times & \\ & \times & \times & & & \\ & \times & \times & \times & \times & \\ & & \times & & & \\ & & \times & \times & \times & \\ & & & \times & \times & \\ & & & & \times & \end{pmatrix}.$$

To create the most useful structure of this form in nonmatched models, some care is needed in ordering the nonlinear variables. In particular, it is helpful for all the parameters that enter a given term to have nearby indices.

10.4. General models. In a general separable model (10.1), each term may depend on more than one nonlinear variable, and a given nonlinear variable may appear in more than one term. This situation occurs in the following model with $\ell = 4$ and $n = 3$:

$$\eta_i(c, x) = c_1 \exp(x_1 t_i) + c_2 x_2^2 t_i + c_3 \exp(x_3^2 x_4 t_i) + c_4 \sin(x_3 x_4 t_i). \tag{10.17}$$

For such models, the structure of the Jacobian is more complicated than shown in (10.14), since Γ_s of (10.8) in general has more than one nonzero column. Nonetheless, since most practical models do not include all nonlinear variables in every term, the Jacobian often displays structure that can be exploited along familiar lines.

If the QR factorization of the Jacobian J is needed, it was suggested in [11] that the QR factorization of a matrix $\tilde{G}$ analogous to G (10.12) should be computed. The matrix $\tilde{G}$ contains the nonzero columns of the n matrices $\{\Gamma_s\}$ of (10.8), where all the nonzero columns from Γ_1 appear first, followed by the nonzero columns from Γ_2, and so on. Let $\tilde{Q}_G$ and $\tilde{R}_G$ be the matrices from the QR factorization of $\tilde{G}$, so that $\tilde{R}_G$ is a $t \times t$ upper triangle, where t is the number of columns of $\tilde{G}$. (We assume that $m > t$.) It follows from the definition of Γ_s that the matrix $\tilde{Q}_G \Gamma_s$ contains a mixture of zero columns and columns of $\tilde{R}_G$ in natural order.

Multiplying the kth Jacobian block $J^{(k)}$ of (10.8) by $\tilde{Q}_G$, we have

$$\tilde{J}^{(k)} = \tilde{Q}_G J^{(k)} = \begin{pmatrix} \tilde{Q}_G \Gamma_1 c^{(k)} & \tilde{Q}_G \Gamma_2 c^{(k)} & \cdots & \tilde{Q}_G \Gamma_n c^{(k)} \end{pmatrix}. \tag{10.18}$$

Because the columns of $\tilde{Q}_G \Gamma_s$ are either zero or columns of $\tilde{R}_G$, the columns of $\tilde{J}^{(k)}$ have a special form. In particular, if x_1 occurs in s_1 terms of the model, column 1 of $\tilde{J}^{(k)}$ contains nonzeros in its first s_1 positions; if x_2 occurs in s_2 terms, column 2 of $\tilde{J}^{(k)}$ contains nonzeros in its first $s_1 + s_2$ positions; and so on.

In the model of (10.17), for example, x_1 and x_2 each occur in one term, and x_3 and x_4 each occur in two terms. Thus $\tilde{G}$ has six columns, and the transformed matrix $\tilde{J}^{(k)}$ of (10.18) has the form

$$\tilde{J}^{(k)} = \begin{pmatrix} \times & \times & \times & \times \\ & \times & \times & \times \\ & & \times & \times \\ & & \times & \times \\ & & & \times \\ & & & \times \end{pmatrix}. \tag{10.19}$$

As long as each nonlinear parameter does not occur in very many terms, the matrix $\tilde{J}^{(k)}$ will contain substantially more zeros than $J^{(k)}$. If the form of the model is known in advance, the structure of $\tilde{J}^{(k)}$ will also be known, and techniques similar to those used in struct1 can be used to obtain the QR factorization of $\tilde{J}$; compare Fig. 4.1 and (10.19).

11. Conclusions. We have shown that exploiting the structure of certain large dense linear least-squares problems leads to substantial improvements in speed and storage compared to using off-the-shelf "black-box" techniques. In particular, we have presented details and tradeoffs for a variety of strategies that take advantage of the structure in a system identification problem (for both real and complex versions) and in separable nonlinear least-squares problems with multiple data sets. We anticipate that similar approaches will be developed for other problems.

Acknowledgment. We thank the referees for several helpful suggestions.

REFERENCES

[1] E. ANDERSON, Z. BAI, C. BISCHOF, J. DEMMEL, J. DONGARRA, J. DUCROZ, A. GREENBAUM, S. HAMMARLING, A. MCKENNEY, S. OSTROUCHOV AND D. SORENSEN, LAPACK *Users' Guide*, Society for Industrial and Applied Mathematics, Philadelphia, 1992.

[2] D. S. BAYARD, *High-order wide-band frequency domain identification using composite curve fitting*, Proc. 1992 Amer. Control Conf., 1992.

[3] ———, *Multivariable frequency domain identification via 2-norm minimization*, Proc. 1992 Amer. Control Conf., 1992.

[4] A. BJÖRCK, *Stability analysis of the method of seminormal equations for linear least-squares problems*, Linear Algebra Appl., 88/89 (1987), pp. 31–48.

[5] C. CHEN, J. JUANG, AND G. LEE, *Frequency domain state-space system identification*, Report TM-107659, NASA, 1992.

[6] J. W. DEMMEL, M. T. HEATH, AND H. A. VAN DER VORST, *Parallel numerical linear algebra*, in Acta Numerica 1993, A. Iserles, ed., Cambridge University Press, New York, 1993, pp. 111–197.

[7] G. H. GOLUB AND R. LEVEQUE, *Extensions and uses of the variable projection algorithm for solving nonlinear least-squares problems*, in Proc. 1979 Army Numerical Analysis and Computers Conference, Army Research Office, 1979.

[8] G. H. GOLUB AND V. PEREYRA, *The differentiation of pseudo-inverses and nonlinear least-squares problems whose variables separate*, SIAM J. Numer. Anal., 10 (1973), pp. 413–432.

[9] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, 2nd ed., Johns Hopkins University Press, Baltimore, MD, 1989.

[10] L. KAUFMAN, *A variable projection method for solving separable nonlinear least-squares problems*, BIT, 15 (1975), pp. 49–57.

[11] L. KAUFMAN AND G. SYLVESTER, *Separable nonlinear least squares with multiple right-hand sides*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 68–89.

[13] F. LEMBREGTS, J. LEURIDAN, AND H. VAN BRUSSEL, *Frequency domain direct parameter identification for modal analysis: state space formulation*, Proc. 7th Internat. Modal Analysis Conf. (1989), pp. 1271–1277.

[14] L. LJUNG, *System Identification: Theory for the User*, Prentice-Hall, London, 1987.

[15] C. Y. SHIH, Y. G. TSUEI, R. J. ALLEMANG, AND D. L. BROWN, *A frequency domain global parameter estimation method for multiple reference frequency response measurements*, Mech. Sys. Signal Proc. (1988), pp. 349–365.

[16] T. SÖDERSTRÖM AND P. STOICA, *System Identification*, Prentice-Hall, London, 1989.

[17] G. S. SYLVESTER, *A frequency domain Prony method*, AT&T Internal Memorandum, 1989.

[18] H. VOLD, *Numerically robust frequency domain modal parameter estimation*, Sound and Vibration (1990), pp. 38–40.

SIAM J. OPTIMIZATION
Vol. 4, No. 4, pp. 872–893, November 1994
© 1994 Society for Industrial and Applied Mathematics
011

SERIAL AND PARALLEL COMPUTATION OF KARUSH–KUHN–TUCKER POINTS VIA NONSMOOTH EQUATIONS*

JONG-SHI PANG†

Abstract. In this paper, a robust, iterative algorithm is presented for computing Karush–Kuhn–Tucker (KKT) points of nonlinear programs. This algorithm is a variation of the NE/SQP method for solving the nonlinear complementarity problem; it makes use of a merit function that combines the original objective function of the nonlinear program with a residual function of the KKT conditions formulated as a system of nonsmooth equations. Global convergence and a Q-quadratic rate of convergence of the algorithm are established under some standard constraint qualifications in nonlinear programming theory, but without the strict complementarity condition. Parallel implementations of the algorithm are discussed.

Key words. nonlinear programs, nonlinear complementarity, nonlinear equations, penalty functions, parallel algorithms

AMS subject classifications. 90C30, 90C33

1. Problem statement.

We consider a nonlinear program formulated in the following form:

$$\begin{array}{lll} \text{minimize} & f(x) & \\ \text{subject to} & x \in C, & h(x) = 0, \end{array} \tag{1}$$

where C is a nonempty polyhedron in R^n, and $f : R^n \to R$ and $h : R^n \to R^q$ are functions satisfying the following blanket differentiability assumption:

A0. f and each h_j are twice continuously differentiable on an open set X containing C; moreover, the second derivative functions $\nabla^2 f, \nabla^2 h_j : X \to R^{n \times n}$ are all bounded on C.

The special feature of the formulation (1) is that all nonlinear constraints of the problem are contained in the equation $h(x) = 0$. This is not a restriction, however, because any (nonlinear) inequality can be trivially converted into an equation by adding (or subtracting) a nonnegative slack variable which is included as a component of the vector x. Throughout this paper, we shall assume that C has the representation

$$C = \{x \in R^n : g(x) \leq 0\},$$

where $g(x) \equiv Ax - b$ for some given matrix $A \in R^{m \times n}$ and vector $b \in R^m$.

Define the Lagrangian function $L : R^n \times R^m_+ \times R^q \to R$ in the usual way: for a triple $z = (x, \lambda, \mu) \in R^n \times R^m_+ \times R^q$,

$$L(z) \equiv f(x) + \lambda^T g(x) + \mu^T h(x).$$

Throughout this paper, z always denotes an appropriate triple $(x, \lambda, \mu) \in R^n \times R^m_+ \times R^q$; we also let $N \equiv n + m + q$ be the dimension of z. The KKT system of (1) can be

* Received by the editors July 14, 1993; accepted for publication (in revised form) April 6, 1994. This work was based on research supported by National Science Foundation grants DDM-9104078 and CCR-92137389.

† Department of Mathematical Sciences, The Johns Hopkins University, Baltimore, Maryland 21218-2689 (jsp@vicp.mts.jhu.edu).

stated as the following system of nonsmooth equations in the variable z:

$$H(z) \equiv \begin{pmatrix} \nabla_x L(z) \\ \min(\lambda, -g(x)) \\ h(x) \end{pmatrix} = 0, \tag{2}$$

where min denotes the componentwise minimum operator on a pair of vectors. The nonsmooth function H, which is a mapping from R^N into itself, plays a central role in the algorithm to be presented later; the zeros of H are precisely the KKT triples of the nonlinear program (1).

For a given scalar $c > 0$, we introduce the penalized merit function

$$P_c(z) = f(x) + \frac{c}{2}\|H(z)\|^2, \qquad z = (x, \lambda, \mu) \in R^N,$$

where $\|\cdot\|$ is the ℓ_2-norm of vectors in R^N. In principle, we could use an arbitrary norm [12] to define this function; nevertheless, as we shall see later, the ℓ_2-norm leads to quadratic programming subproblems that can be solved by a host of highly effective parallel algorithms, thus enabling the design of some parallel iterative methods for computing a KKT point of (1). Let

$$\theta(z) = \tfrac{1}{2}\|H(z)\|^2, \qquad z \in R^N. \tag{3}$$

This function $\theta(z)$ constitutes the major difference between the penalty function $P_c(z)$ and many commonly used penalty functions for the nonlinear program (1); see [3] and the extensive references therein. More specifically, in $P_c(z)$, the entire set of KKT conditions is penalized through the function $\theta(z)$; whereas in most other penalty functions, only the constraints of (1) are penalized.

The approach we take to compute a KKT triple of the given program (1) is to find a zero of H by decreasing the function $P_c(z)$ for certain values of c. This (somewhat heuristic) approach is an extension of the ideas used in the NE/SQP method for solving the nonlinear complementarity problem [20] and the arbitrary-norm trust region method for solving nonsmooth equations [12]. (References on solving equations of the Gauss–Newton type include [10], [21], and [25].) In the methods described in [20] and [12], the function $\theta(z)$ was the merit function and the fact that the objective function $f(x)$ should be minimized was ignored; thus, there was generally no guarantee that the function $f(x)$ would be decreased. This could endanger the successful computation of a local minimizer of (1) when this is not a convex program. The use of the merit function P_c partially atones for this drawback of the equation approach to computing a KKT point of (1). Nevertheless, as the two referees of this paper correctly pointed out, the use of P_c still generally does not guarantee that the function f would be decreased. As it presently stands, the strength of the proposed algorithm when applied to a nonconvex program lies in its theoretical properties. Indeed, the global convergence of the resulting algorithm and its local, superlinear rate of convergence (quadratic rate with a Lipschitzian assumption) will be established under some fairly standard constraint qualifications in nonlinear programming theory, but without the strict complementarity condition. A side benefit of using P_c as the merit function is the removal of the so-called Maratos effect that has been a detriment to the rapid local convergence of many sequential quadratic programming (SQP) algorithms.

The algorithm described in this paper for computing a KKT point of (1) has the potential of being implementable on parallel computers. The resulting parallel

procedures are based on a concept that is fundamentally different from many existing parallel algorithms for solving special cases of (1) which typically pertain to this problem being a convex program [1], [9]. In a nutshell, the parallel schemes proposed in this paper are developed within an SQP framework; the parallelism occurs in the solution of the quadratic programming subproblems. There are several benefits from this approach for designing parallel methods for solving (1): (i) The fact that the global convergence of the methods is not restricted to the class of convex programs; (ii) A superlinear (and even quadratic) rate of convergence can be established for the sequence of primary iterates. From a theoretical point of view, this rapid local convergence compares favorably with many other schemes (such as those described in the references) that typically achieve at best a linear rate; (iii) Since the quadratic programming subproblems are strictly convex, there is great flexibility in their solution by parallel algorithms. Later in this paper, we describe one particular parallel implementation of the sequential algorithm developed in §3.

2. The quadratic subprogram. As in [20], [12], we can use either a line search or trust region approach to design our (sequential) algorithm. In this paper we have chosen the former.

The algorithm to be developed is an iterative procedure that starts at an arbitrary triple $z^0 = (x^0, \lambda^0, \mu^0) \in Z$, where

$$Z \equiv C \times R^m_+ \times R^q \subseteq R^N,$$

and generates a sequence of vectors $\{z^k\}$ that remains in the set Z. Note that the sequence $\{x^k\}$ is contained in the set C but these vectors are not necessarily feasible to (1). The generation of each iterate z^{k+1} consists of two computational steps: (i) direction generation and (ii) step size determination. The former step involves solving a (strictly) convex quadratic program. In what follows, we discuss the essential properties of this quadratic program.

For a given triple $z = (x, \lambda, \mu) \in Z$, define two complementary index subsets of $\{1, \ldots, m\}$:

$$I(z) = \{i : \lambda_i + g_i(x) > 0\} \quad \text{and} \quad J(z) = \{i : \lambda_i + g_i(x) \leq 0\}.$$

We also define a function $\phi(z, dz)$ of two arguments: for $z = (x, \lambda, \mu) \in Z$ and $dz = (dx, d\lambda, d\mu) \in R^N$, let

$$\phi(z, dz) = \tfrac{1}{2}(T_1 + T_2 + T_3 + T_4),$$

where

$$T_1 = \|\nabla_x L(z) + \nabla^2_{xx} L(z)dx + \textstyle\sum_{i=1}^m \nabla g_i(x) d\lambda_i + \sum_{j=1}^q \nabla h_j(x) d\mu_j\|^2,$$

$$T_2 = \|h(x) + \nabla h(x) dx\|^2, \qquad T_3 = \textstyle\sum_{i \in I(z)} \left(g_i(x) + \nabla g_i(x)^T dx\right)^2,$$

$$T_4 = \textstyle\sum_{i \in J(z)} (\lambda_i + d\lambda_i)^2.$$

This function $\phi(z, dz)$ is related to the directional derivative of the function θ in (3) at the point z along the direction dz. To explain this and other relations between these two functions, we write

$$\phi_z(dz) \equiv \phi(z, dz), \qquad dz \in R^N$$

to signify that $z \in Z$ is fixed and only dz varies. We have the following result.

LEMMA 2.1. *For a given vector $z \in Z$, the following properties hold:*

(a) $\phi(z,0) = \theta(z)$;

(b) $\phi(z,\cdot)$ *is a convex quadratic function; thus, for every* $\tau \in [0,1]$ *and any* $dz \in R^N$,

$$\phi(z,\tau dz) - \theta(z) \leq \tau(\phi(z,dz) - \theta(z)); \tag{4}$$

(c) $\limsup_{dz\to 0} \frac{\theta(z+dz)-\phi(z,dz)}{\|dz\|} \leq 0$;

(d) $\theta'(z,dz) \leq \phi'_z(0,dz)$ *for all* $dz \in R^N$.

Proof. Properties (a) and (b) are obvious. Property (c) can be proved by examining the individual summands that comprise $\theta(z+dz)$ and comparing them with the respective terms in $\phi(z,dz)$; for further details, we refer the reader to [12] where a more general result was established for an arbitrary norm. Finally, (d) is an immediate consequence of (c) and (a). □

Subsequently, we need a result that is a sequential version of Lemma 2.1(c). We state this result below.

LEMMA 2.2. *Let* $\{z^k\}$ *and* $\{dz^k\}$ *be sequences of vectors and* $\{\tau_k\}$ *be a sequence of positive scalars satisfying the properties*: (i) *for each* k, z^k *and* $z^k + dz^k$ *belong to* Z; (ii) $\{\theta(z^k)\}$, $\{\phi(z^k,dz^k)\}$, *and* $\{dz^k\}$ *are bounded*; *and* (iii) $\{\tau_k\}$ *converges to zero. Then under assumption* A0,

$$\limsup_{k\to\infty} \frac{\theta(z^k+\tau_k dz^k) - \phi(z^k,\tau_k dz^k)}{\|\tau_k\|} \leq 0.$$

Proof. By conditions (i) and (iii), we may assume without loss of generality that for each k, $z^k + \tau_k dz^k \in Z$. The proof of the desired inequality is again made by comparing the individual summands that comprise $\theta(z^k + \tau_k dz^k)$ and $\phi(z^k,\tau_k dz^k)$ and using differentiability assumption A0 and boundedness assumption (ii). We omit the details. □

For a given vector $z \in Z$ and a symmetric positive definite matrix $B \in R^{N\times N}$, let QP (z,B) denote the following strictly convex quadratic program in the variable $dz \in R^N$:

$$\begin{aligned} &\text{minimize} && \phi_z(dz) + \tfrac{1}{2} dz^T B dz \\ &\text{subject to} && z + dz \in Z. \end{aligned} \tag{5}$$

The problem QP (z,B) is central to the line search method to be described later. It is different from the quadratic subprograms that many SQP methods solve [14], [22], [5], [4]. Unlike these subprograms, QP (z,B) is derived from the function $H(z)$ in (2) which originates from the nonlinear complementarity problem and variational inequality; see [20], [19]. In particular, there is an easy generalization of our approach to treat the variational inequality; but the SQP methods mentioned in the cited references are not applicable for solving the latter problem.

The following result summarizes some useful properties of the quadratic program QP (z,B).

LEMMA 2.3. *Let* $z \in Z$ *be given and* $B \in R^{N\times N}$ *be a symmetric positive definite matrix. The following statements hold for the problem QP* (z,B).

(a) *There always exists a unique optimal solution* $d\bar{z}$ *to QP* (z,B) *which satisfies*

$$\phi(z,d\bar{z}) \leq \phi_z(d\bar{z}) + \tfrac{1}{2} d\bar{z}^T B d\bar{z} \leq \theta(z). \tag{6}$$

(b) *Let* $\bar{\gamma} \equiv \phi(z,d\bar{z})$. *Then* $\bar{\gamma} = \theta(z)$ *if and only if* $d\bar{z} = 0$.

(c) *If* $\bar{\gamma} < \theta(z)$ *and* c *is any positive scalar satisfying*

$$c(\theta(z) - \bar{\gamma}) - \nabla f(x)^T d\bar{x} \geq c_0(\theta(z) - \bar{\gamma}) \tag{7}$$

for some constant $c_0 > 0$, *then for every scalar* $\sigma \in (0,1)$, *there exists a scalar* $\bar{\tau} > 0$ *such that for all* $\tau \in [0, \bar{\tau}]$,

$$P_c(z + \tau d\bar{z}) - P_c(z) \leq -\sigma\tau c_0(\theta(z) - \bar{\gamma}).$$

Proof. The vector $dz = 0$ is clearly a feasible solution to QP (z, B). Since the function $\phi_z(dz) + \frac{1}{2} dz^T B dz$ is strongly convex in dz, QP (z, B) has a unique optimal solution $d\bar{z}$. The expression (6) holds by the observation made in the opening statement of the proof. This establishes (a). Statement (b) is an immediate consequence of the uniqueness of the optimal solution $d\bar{z}$.

To prove statement (c), suppose (7) holds and its right-hand side is positive. We have

$$\begin{aligned}
&P_c(z + \tau d\bar{z}) - P_c(z) = f(x + \tau d\bar{x}) - f(x) + c(\theta(z + \tau d\bar{z}) - \theta(z)) \\
&= \tau \nabla f(x)^T d\bar{x} + o(\tau) + c(\theta(z + \tau d\bar{z}) - \phi(z, \tau d\bar{z})) + c(\phi(z, \tau d\bar{z}) - \theta(z)) \\
&\leq \tau(\nabla f(x)^T d\bar{x} + c(\phi(z, d\bar{z}) - \theta(z))) + o(\tau) \quad \text{by (b) and (c) of Lemma 2.1} \\
&\leq -\tau c_0(\theta(z) - \bar{\gamma}) + o(\tau) \quad \text{by (7)} \\
&\leq -\sigma\tau c_0(\theta(z) - \bar{\gamma}) \quad \text{for all } \tau > 0 \text{ sufficiently small,}
\end{aligned}$$

where the last inequality holds because $\theta(z) > \bar{\gamma}$ by assumption. This completes the proof. □

The above lemma does not address the case where $d\bar{z} = 0$. To treat this case, we introduce several concepts. First, we define two disjoint subsets of $J(z)$:

$$K_0(z) = \{i \in J(z) : \lambda_i = -g_i(x) = 0\}, \qquad K_+(z) = \{i \in J(z) : \lambda_i = -g_i(x) > 0\}.$$

Notice that if z is a KKT triple of (1), then $K_+(z) = \emptyset$; nevertheless, this is not necessarily the case for an arbitrary vector $z \in Z$. With these index sets, we define the following quadratic program in the variable $v \in R^n$:

$$\begin{aligned}
&\text{minimize} && \nabla f(x)^T v + \tfrac{1}{2} v^T \nabla^2_{xx} L(z) v \\
&\text{subject to} && h_j(x) + \nabla h_j(x)^T v = 0 && \text{for all } j = 1, \ldots, q, \\
&&& g_i(x) + \nabla g_i(x)^T v = 0 && \text{for all } i \in I(z), \\
&&& g_i(x) + \nabla g_i(x)^T v \leq 0 && \text{for all } i \in K_0(z), \\
&&& g_i(x) + \nabla g_i(x)^T v \geq 0 && \text{for all } i \in K_+(z).
\end{aligned} \tag{8}$$

Note that the constraints in this problem do not contain those that correspond to the indices $i \in J(z) \setminus (K_0(z) \cup K_+(z))$.

DEFINITION 1. *A triple* $z = (x, \lambda, \mu) \in Z$ *is said to be* s-regular *if* (8) *has a stationary point (in particular, it must be feasible).*

Associated with the constraints in the quadratic program (8), we define the polyhedral cone

$$\begin{aligned}
\mathcal{C}(z) = \{v \in R^n : \ & \nabla h_j(x)^T v = 0 && \text{for all } j = 1, \ldots, q, \\
& \nabla g_i(x)^T v = 0 && \text{for all } i \in I(z), \\
& \nabla g_i(x)^T v \leq 0 && \text{for all } i \in K_0(z), \\
& \nabla g_i(x)^T v \geq 0 && \text{for all } i \in K_+(z)\}.
\end{aligned}$$

If z is a KKT triple of the nonlinear program (1), then $K_+(z) = \emptyset$ and the cone $\mathcal{C}(z)$ reduces to the *critical cone* of this problem at z [23]. In essence, we have extended the concept of the critical cone, which was previously defined only at a KKT point, to an arbitrary triple $z \in Z$.

A sufficient condition for a given triple $z \in Z$ to be s-regular is that (i) the quadratic program (8) is feasible and (ii) the Hessian matrix $\nabla^2_{xx} L(z)$ is *strictly copositive* on the cone $\mathcal{C}(z)$, i.e.,

$$0 \neq v \in \mathcal{C}(z) \quad \Rightarrow \quad v^T \nabla^2_{xx} L(z) v > 0.$$

This follows from a basic result in quadratic programming theory; see [8]. Indeed, under these two assumptions, (8) has a global minimum that must necessarily be a stationary point. In turn, a sufficient condition for (8) to be feasible is that the triple $z \in Z$ satisfies the following two conditions.

S1. The gradient vectors

$$\{\nabla h_j(x) : \text{ all } j\} \cup \{\nabla g_i(x) : \text{ all } i \in I(z)\}$$

are linearly independent.

S2. There exists a vector $\tilde{v} \in R^n$ such that

$$\begin{aligned}
\nabla h_j(x)^T \tilde{v} &= 0 \quad \text{for all } j = 1, \ldots, q, \\
\nabla g_i(x)^T \tilde{v} &= 0 \quad \text{for all } i \in I(z), \\
\nabla g_i(x)^T \tilde{v} &< 0 \quad \text{for all } i \in K_0(z), \\
\nabla g_i(x)^T \tilde{v} &> 0 \quad \text{for all } i \in K_+(z).
\end{aligned}$$

Indeed, under these conditions, it follows from a standard theorem of the alternatives [17] that the system of linear inequalities in the variable $v \in R^n$:

$$\begin{aligned}
s_j + \nabla h_j(x)^T v &= 0 \quad \text{for all } j = 1, \ldots, q, \\
r_i + \nabla g_i(x)^T v &= 0 \quad \text{for all } i \in I(z), \\
r_i + \nabla g_i(x)^T v &\leq 0 \quad \text{for all } i \in K_0(z), \\
r_i + \nabla g_i(x)^T v &\geq 0 \quad \text{for all } i \in K_+(z)
\end{aligned}$$

must be consistent for all vectors $s \in R^q$ and $r \in R^m$. In particular, (8) is then feasible.

If z happens to be a KKT triple of the nonlinear program (1), then the two conditions S1 and S2 are precisely the *strict Mangasarian–Fromovitz constraint qualification* (SMFCQ) at z. Thus, similar to the definition of the critical cone, we have extended this constraint qualification, which was previously defined only as a KKT triple, to an arbitrary triple $z \in Z$. As demonstrated in [15], the SMFCQ at a KKT triple (x, λ, μ) is necessary and sufficient for the multipliers (λ, μ) to be unique corresponding to the point x. In particular, this constraint qualification is implied by the linear independence constraint qualification (LICQ) that requires the gradients of all active constraints to be linearly independent; see §5 for the definition of the latter constraint qualification.

With the above preparation, we are ready to state an important result pertaining to the case $\bar{\gamma} = \theta(z)$ in the QP (z, B).

THEOREM 2.4. *Let $z \in Z$ be given and $B \in R^{N \times N}$ be a symmetric positive definite matrix. Suppose that $d\bar{z} = 0$ is the unique optimal solution of QP (z, B). The following statements are equivalent:*

(A) $\theta(z) = 0$;

(B) z *is a* KKT *triple of* (1);

(C) z *is an* s-regular *triple.*

Moreover, if $\nabla^2_{xx} L(z)$ *is strictly copositive on the cone* $\mathcal{C}(z)$*, then each of the statements* (A), (B), *and* (C) *is further equivalent to the following.*

(D) $h(x) = 0$, $g_i(x) = 0$ *for all* $i \in I(z)$, *and* $K_+(z) = \emptyset$.

In this case, x *is a strict local minimum of* (1) *if any one of the four conditions* (A), (B), (C), *or* (D) *holds. Finally if, in addition, the two conditions* S1 *and* S2 *hold at* z*, then*

(E) *all four statements* (A), (B), (C), *and* (D) *are valid*;

(F) (λ, μ) *is the unique* KKT *multiplier pair associated with* x; *and*

(G) z *is an isolated* KKT *triple of* (1) *and* x *is an isolated local minimum of the same program.*

Proof. (A) $\Rightarrow$ (B). The proof of this theorem is obvious.

(B) $\Rightarrow$ (C). The zero vector is obviously a stationary point of (8) if z is a KKT triple of (1).

(C) $\Rightarrow$ (A). The proof is by contradiction. Suppose $\theta(z) > 0$. Let $\bar{v}$ be a stationary point of (8). Then there exist multipliers $\bar{w}$ and $\bar{u}$ such that

$$\nabla f(x) + \nabla^2_{xx} L(z)\bar{v} + \sum_{j=1}^{q} \bar{w}_j \nabla h_j(x) + \sum_{i \in I(z) \cup K_0(z)} \bar{u}_i \nabla g_i(x) - \sum_{i \in K_+(z)} \bar{u}_i \nabla g_i(x) = 0 \tag{9}$$

$$\bar{u}_i \geq 0, \quad \bar{u}_i (g_i(x) + \nabla g_i(x)^T \bar{v}) = 0 \quad \text{for all } i \in K_0(z) \cup K_+(z).$$

We claim that for all $\tau \in (0,1)$ sufficiently small, the vector $dz = (dx, d\lambda, d\mu)$ defined by

$$dx = \tau \bar{v}, \qquad d\mu = \tau(\bar{w} - \mu), \tag{10}$$

$$d\lambda_i = \begin{cases} \tau(\bar{u}_i - \lambda_i) & \text{if } i \in I(z) \cup K_0(z), \\ -\tau(\bar{u}_i + \lambda_i) & \text{if } i \in K_+(z), \\ -\tau\lambda_i & \text{if } i \in J(z) \setminus (K_0(z) \cup K_+(z)) \end{cases} \tag{11}$$

is feasible to QP (z, B) and

$$\phi_z(dz) \leq (1-\tau)^2 \theta(z) + O(\tau^2). \tag{12}$$

We verify the latter inequality by examining the individual summands of $\phi_z(dz)$ and along the way demonstrate the feasibility of dz to QP (z, B). Note that the latter feasibility amounts to two requirements: $x + dx \in C$ and $\lambda + d\lambda \geq 0$.

By (9), we can easily verify

$$\nabla_x L(z) + \nabla^2_{xx} L(z) dx + \sum_{i=1}^{m} d\lambda_i \nabla g_i(x) + \sum_{j=1}^{q} d\mu_j \nabla h_j(x) = (1-\tau)\nabla_x L(z),$$

which implies

$$\left\| \nabla_x L(z) + \nabla^2_{xx} L(z) dx + \sum_{i=1}^{m} d\lambda_i \nabla g_i(x) + \sum_{j=1}^{q} d\mu_j \nabla h_j(x) \right\| = (1-\tau) \|\nabla_x L(z)\|. \tag{13}$$

Similarly, we have

$$\|h(x) + \nabla h(x)dx\| = (1 - \tau)\|h(x)\|. \tag{14}$$

Moreover, for all $i \in I(z)$, we have $\lambda_i > -g_i(x) \geq 0$ and

$$g_i(x) + \nabla g_i(x)^T dx = (1 - \tau)g_i(x) \leq 0, \tag{15}$$

$$\lambda_i + d\lambda_i = (1 - \tau)\lambda_i + \tau\bar{u}_i \geq 0, \tag{16}$$

provided that $\tau > 0$ is sufficiently small. Next, consider an index $i \in J(z)$. There are three cases: (i) $i \in K_0(z)$, (ii) $i \in K_+(z)$, and (iii) $i \in J(z) \setminus (K_0(z) \cup K_+(z))$. Consider the first case where $i \in K_0(z)$. We have

$$g_i(x) + \nabla g_i(x)^T dx \leq (1 - \tau)g_i(x) = 0, \tag{17}$$

and since $\lambda_i = 0$ for $i \in K_0(z)$,

$$\lambda_i + d\lambda_i = \tau\bar{u}_i \geq 0. \tag{18}$$

Consider the second case where $i \in K_+(z)$. Then $g_i(x) < 0$; thus for $\tau > 0$ sufficiently small, we have

$$g_i(x) + \nabla g_i(x)^T dx = g_i(x) + \tau\nabla g_i(x)^T\bar{v} \leq 0. \tag{19}$$

Moreover, since $\bar{u}_i \geq 0$,

$$0 \leq \lambda_i + d\lambda_i = (1 - \tau)\lambda_i - \tau\bar{u}_i \leq (1 - \tau)\lambda_i, \tag{20}$$

where the first inequality holds because $\lambda_i > 0$ and τ is sufficiently small. Finally, consider an index i such that $\lambda_i + g_i(x) < 0$. Then $g_i(x)$ is negative. Consequently, for $\tau > 0$ sufficiently small, we have

$$g_i(x) + \nabla g_i(x)^T dx = g_i(x) + \tau\nabla g_i(x)^T\bar{v} \leq 0, \tag{21}$$

$$\lambda_i + d\lambda_i = (1 - \tau)\lambda_i \geq 0. \tag{22}$$

From (15)—(22), we see that the vector dz is feasible to QP (z, B); moreover,

$$\sum_{i \in I(z)} (g_i(x) + \nabla g_i(x)^T dx)^2 = (1 - \tau)^2 \sum_{i \in I(z)} g_i(x)^2,$$

$$\sum_{i \in J(z)} (\lambda_i + d\lambda_i)^2 \leq (1 - \tau)^2 \sum_{i \in J(z)} \lambda_i^2 + O(\tau^2).$$

These last two expressions, together with (13) and (14), easily yield (12).

Consequently, it follows that

$$(1 - 2\tau)\theta(z) + O(\tau^2) \geq \phi_z(dz) + \tfrac{1}{2}(dz)^T B dz \geq \theta(z)$$

for all $\tau > 0$ sufficiently small. Simplifying and letting $\tau \downarrow 0$, we conclude that $\theta(z) = 0$; thus (A) holds.

If (D) holds, then (8) is feasible. Consequently, by the discussion preceding the statement of this theorem, it follows that if $\nabla^2_{xx} L(z)$ is strictly copositive on $\mathcal{C}(z)$, then (8) has a stationary point. Hence, (D) implies (C) under this copositivity assumption. Conversely, (B) clearly implies (D). If any one of the four statements (A)–(D) holds, then the strict copositivity assumption of $\nabla^2_{xx} L(z)$ becomes the well-known second-order sufficiency condition at the KKT triple z; hence, the desired claim about x follows from a standard result (see, e.g., [3]).

Finally, (E) and (F) follow from the same discussion given before the statement of this theorem. The isolatedness of z as a KKT triple of (1) is a consequence of recent sensitivity results proved in [13]; from this conclusion, the isolatedness of x as a local minimum of the same program follows readily. □

3. Description of the algorithm. The proposed algorithm for computing a KKT point of (1) is a robust, line search method that consists of solving convex quadratic programming subproblems to generate descent directions for the merit function P_c. Although our algorithm is similar in spirit to some recent algorithms of this kind (see, e.g., [2], [5], [4]), there are certain differences as well (e.g., in the merit functions). It is not our intention in this paper to compare these algorithms; instead, we describe our algorithm, establish its convergence properties, and discuss possible ways for its parallel implementation.

The following are the main steps of the promised algorithm for computing a KKT triple of (1).

Step 0 (Initialization). Let $\rho, \sigma \in (0,1)$ be given. Let $z^0 = (x^0, \lambda^0, \mu^0) \in Z$ be an arbitrary vector and $c_0 > 0$ be an arbitrary scalar. Let B_0 be a given symmetric positive definite matrix. Set $k = 0$.

Step 1 (Direction generation). Given the iterate $z^k \in Z$, the symmetric positive definite matrix B_k, and the positive scalar c_k, solve the QP (z^k, B_k) and let dz^k be the unique optimal solution obtained. If $\gamma_k \equiv \phi(z^k, dz^k) = \theta(z^k)$, stop. Otherwise, determine a scalar $c_{k+1} \geq c_k$ such that

$$c_{k+1}(\theta(z^k) - \gamma_k) - \nabla f(x^k)^T dx^k \geq c_0(\theta(z^k) - \gamma_k). \tag{23}$$

Step 2 (Armijo line search). Let m_k be the smallest nonnegative integer m such that

$$P_{c_{k+1}}(z^k + \rho^m dz^k) - P_{c_{k+1}}(z^k) \leq -\sigma \rho^m c_0(\theta(z^k) - \gamma_k). \tag{24}$$

Set $z^{k+1} = z^k + \rho^{m_k} dz^k$. (Note that z^{k+1} remains in Z.)

Step 3 (Termination check). If z^{k+1} satisfies a prescribed termination rule (such as $\theta(z^{k+1}) \leq \varepsilon$ for some given tolerance $\varepsilon > 0$); stop. Otherwise, choose a symmetric positive definite matrix B_{k+1} and return to Step 1 with k replaced by $k+1$.

According to Lemma 2.3, if $\gamma_k < \theta(z^k)$, then the integer m_k in Step 2 can be determined in a finite number of trials, starting with the value of $m = 0$ and increasing it by one at each trial. Consequently, each iterate z^{k+1} is well defined. In the next section, we show that if the algorithm does not terminate in Step 2, then under certain conditions to be specified, it must terminate in Step 3 in a finite number of iterations.

Notice that unlike many SQP methods that update the penalty parameter c via the multipliers of the constraints in the quadratic programming subproblems [14], [22], [5], [4], we rely on the inequality (23) to determine c_{k+1}.

4. Subsequential convergence. Assume that the algorithm described in §3 generates an infinite sequence of iterates $\{z^k\} \subseteq Z$ along with the sequence of optimal solutions $\{dz^k\}$ to the QP subprograms. By the infinite nature of these sequences, we have $\gamma_k < \theta(z^k)$ for all k. For the subsequent analysis, we need two assumptions:

A1. The objective function $f(x)$ is bounded below on the polyhedron C;

A2. $\sum_{i=0}^{\infty}(c_{i+1} - c_i)\theta(z^i) < \infty$.

Assumption A1 is quite reasonable; it is trivially satisfied if C is bounded. Assumption A2 is a restriction on the growth of the sequence of penalty scalars $\{c_k\}$; it is clearly satisfied if $c_{k+1} = c_k$ for all large k. Incidentally, the latter finiteness condition on the sequence $\{c_k\}$ is a common assumption made in the convergence analysis of SQP methods for constrained optimization problems; see, e.g., [4]. Our analysis relies on the more general assumption A2. In preparation for the subsequent results, we first state a lemma that is related to A2.

LEMMA 4.1. *Let $\{\alpha_k\}$ and $\{\delta_k\}$ be two sequences of scalars satisfying:*

(a) $\alpha_{k+1} \leq \alpha_k + \delta_k$, *for all* $k \geq 0$;

(b) $\{\alpha_k\}$ *is bounded below; and*

(c) *each* δ_k *is nonnegative and* $\sum_{k=0}^{\infty}\delta_k < \infty$.

Then $\lim_{k\to\infty}\alpha_k$ *exists.*

Proof. Let $\alpha'_k = \alpha_k - \inf_\ell \alpha_\ell$ for all k. Then $\{\alpha'_k\}$ is a sequence of nonnegative scalars satisfying $\alpha'_{k+1} \leq \alpha'_k + \delta_k$ for all $k \geq 0$. Thus by (c), it follows that $\{\alpha'_k\}$ is bounded. Moreover, by an easy argument, (see [6, Lemma 2.2.12] for details), it can be shown that $\{\alpha'_k\}$ has a unique accumulation point. Thus, $\lim_{k\to\infty}\alpha'_k$, and hence $\lim_{k\to\infty}\alpha_k$, exists. □

Returning to the analysis of the algorithm, we state the following preliminary result.

LEMMA 4.2. *Under assumptions* A0, A1, *and* A2, *the following properties hold:*

(i) $\limsup_{k\to\infty} c_k\theta(z^k)$ *and* $\limsup_{k\to\infty}\theta(z^k)$ *are both finite;*

(ii) $\lim_{k\to\infty} P_{c_k}(z^k)$ *exists; and*

(iii) $\lim_{k\to\infty}\tau_k(\gamma_k - \theta(z^k)) = 0$ *where* $\tau_k = \rho^{m_k}$.

Proof. By assumption A1, we have

$$f(x^k) \geq \beta \quad \text{for all } k$$

for some constant β. By (24) and the fact that $\gamma_k < \theta(z^k)$, we can show inductively that

$$\begin{aligned}\beta + c_{k+1}\theta(z^{k+1}) &\leq f(x^{k+1}) + c_{k+1}\theta(z^{k+1}) \leq f(x^k) + c_{k+1}\theta(z^k)\\ &= f(x^k) + c_k\theta(z^k) + (c_{k+1} - c_k)\theta(z^k)\\ &\leq \cdots \leq f(x^0) + c_0\theta(z^0) + \textstyle\sum_{i=0}^{k}(c_{i+1} - c_i)\theta(z^i).\end{aligned}$$

Consequently, $\{c_{k+1}\theta(z^{k+1})\}$ is bounded. Since $c_{k+1} \geq c_0 > 0$ for all k, $\{\theta(z^{k+1})\}$ is also bounded. Thus, (i) holds. Moreover, the above proof shows that Lemma 4.1 can be applied to the sequences $\{P_{c_k}(z^k)\}$ and $\{(c_{k+1} - c_k)\theta(z^k)\}$; the conclusion (ii) follows readily. Since

$$P_{c_{k+1}}(z^{k+1}) - P_{c_k}(z^k) = P_{c_{k+1}}(z^{k+1}) - P_{c_{k+1}}(z^k) + (c_{k+1} - c_k)\theta(z^k),$$

it follows from (ii) and assumption A2 that

$$\lim_{k\to\infty}(P_{c_{k+1}}(z^{k+1}) - P_{c_{k+1}}(z^k)) = 0.$$

This easily implies (iii) in view of (24). □

Next, we establish a strengthening of property (iii) in the above lemma by making use of the following additional assumption.

A3. There exist positive scalars α_1 and α_2 such that for all k and all $dz \in R^N$:

$$\alpha_1 \|H(z^k)\| \, \|dz\|^2 \leq dz^T B_k dz \leq \alpha_2 \|H(z^k)\| \, \|dz\|^2.$$

We also recall the boundedness of $\nabla^2 f$ on C as implied by assumption A0. Incidentally, assumption A3 is the only condition we impose on the sequence of matrices $\{B_k\}$.

PROPOSITION 4.3. *Under assumptions* A0–A3, *it holds that*

$$\lim_{k\to\infty} (\gamma_k - \theta(z^k)) = 0.$$

Proof. Since $\limsup_{k\to\infty} \theta(z^k)$ is finite and $0 \leq \gamma_k < \theta(z^k)$, it follows that the sequence $\{\gamma_k - \theta(z^k)\}$ is bounded. It suffices to show that this sequence has a unique accumulation point, namely, zero. Consider any convergent subsequence $\{\gamma_k - \theta(z^k) : k \in \kappa\}$. If

$$\liminf_{k(\in\kappa)\to\infty} \theta(x^k) = 0,$$

then the limit of $\{\gamma_k - \theta(z^k) : k \in \kappa\}$ is zero. So suppose that

$$\liminf_{k(\in\kappa)\to\infty} \theta(x^k) > 0. \tag{25}$$

Since for each k,

$$(dz^k)^T B_k dz^k \leq 2(\theta(z^k) - \gamma_k),$$

assumption A3 and (25) imply that the sequence $\{dz^k : k \in \kappa\}$ is bounded. By part (iii) of Lemma 4.2, it suffices to consider the case where

$$\liminf_{k(\in\kappa)\to\infty} \tau_k = 0.$$

By working with a further subsequence if necessary, we may assume without loss of generality that

$$\lim_{k(\in\kappa)\to\infty} \tau_k = 0,$$

which implies

$$\lim_{k(\in\kappa)\to\infty} m_k = \infty.$$

By the definition of m_k, we deduce that for all $k \in \kappa$ sufficiently large,

$$P_{c_{k+1}}(z^k + \tau'_k dz^k) - P_{c_{k+1}}(z^k) > -\sigma\tau'_k c_0(\theta(z^k) - \gamma_k),$$

where $\tau'_k = \rho^{m_k - 1}$. Substituting the definition of the penalty function, letting $\tilde{z}^k = z^k + \tau'_k dz^k$, and rearranging terms, we deduce

$$\frac{f(\tilde{x}^k) - f(x^k) + c_{k+1}(\theta(\tilde{z}^k) - \theta(z^k))}{\tau'_k} > -\sigma c_0(\theta(z^k) - \gamma_k). \tag{26}$$

Since $\tilde{x}^k$ and x^k belong to C which is convex, by a mean value theorem and assumption A0, we may write

$$f(\tilde{x}^k) - f(x^k) = \tau_k' \nabla f(x^k)^T dx^k + o(\tau_k');$$

moreover, by Lemmas 2.1(b) and 2.2, we have

$$\begin{aligned}\theta(\tilde{z}^k) - \theta(z^k) &\leq \phi(z^k, \tau_k' dz^k) - \theta(z^k) + o(\tau_k') \\ &\leq \tau_k'(\phi(z^k, dz^k) - \theta(z^k)) + o(\tau_k') = \tau_k'(\gamma_k - \theta(z^k)) + o(\tau_k').\end{aligned}$$

Consequently, we deduce

$$\frac{f(\tilde{x}^k) - f(x^k) + c_{k+1}(\theta(\tilde{z}^k) - \theta(z^k))}{\tau_k'} \leq \nabla f(x^k)^T dx^k + c_{k+1}(\gamma_k - \theta(z^k)) + \frac{o(\tau_k')}{\tau_k'}$$

which yields, by (7),

$$\frac{f(\tilde{x}^k) - f(x^k) + c_{k+1}(\theta(\tilde{z}^k) - \theta(z^k))}{\tau_k'} \leq -c_0(\theta(z^k) - \gamma_k) + \frac{o(\tau_k')}{\tau_k'}.$$

Taking limits as $k(\in \kappa) \to \infty$ and recalling (26) and the assumption that the sequence $\{\gamma_k - \theta(z^k) : k \in \kappa\}$ converges, we obtain

$$-\sigma c_0 \lim_{k(\in\kappa)\to\infty} (\theta(z^k) - \gamma_k) \leq -c_0 \lim_{k(\in\kappa)\to\infty} (\theta(z^k) - \gamma_k) \leq 0,$$

which yields (since $\sigma \in (0,1)$),

$$\lim_{k(\in\kappa)\to\infty} (\theta(z^k) - \gamma_k) = 0$$

as desired. □

Our next goal is to study the existence and properties of an accumulation point of the sequence $\{z^k\}$. The argument for the existence of an accumulation point of this sequence proceeds as follows. We impose a boundedness assumption on the level sets of f on C which is sufficient to yield the boundedness of the x-sequence $\{x^k\}$. Then with a certain condition on an accumulation point of $\{x^k\}$, we show that a corresponding subsequence of the multipliers $\{(\lambda^k, \mu^k)\}$ is bounded. To explain the latter condition, let x^* be the limit of any convergent subsequence $\{x^k : k \in \kappa\}$. Define the index set

$$\mathcal{I}(x^*) = \{i : i \in I(z^k) \text{ for infinitely many } k \in \kappa\}.$$

We need the following properties on x^*.

M1. The gradient vectors

$$\{\nabla h_j(x^*) : j = 1, \ldots, q\}$$

are linearly independent.

M2. There exists a vector $\hat{v} \in R^n$ such that

$$\begin{aligned}\nabla h_j(x^*)^T \hat{v} &= 0 \quad j = 1, \ldots, q, \\ \nabla g_i(x^*)^T \hat{v} &< 0 \quad i \in \mathcal{I}(x^*).\end{aligned}$$

It is not hard to recognize the similarity between these two conditions and the famous Mangasarian–Fromovitz constraint qualification at a KKT point [18]; nevertheless, we caution the reader that the vector x^* is not yet known to be a KKT point of (1). In other words, like the concept of the critical cone and the SMFCQ, we have once again considered a well-known condition in nonlinear programming generalized to our algorithmic context.

PROPOSITION 4.4. *Under assumptions* A0–A3, *the following statements hold.*

(i) *If f has bounded level sets on C, then $\{x^k\}$ is bounded.*

(ii) *If x^* is the limit of $\{x^k : k \in \kappa\}$ satisfying the two conditions* M1 *and* M2, *then $\{(\lambda^k, \mu^k) : k \in \kappa\}$ is bounded.*

Proof. By part (ii) of Lemma 4.2, there exists a scalar α such that for all k,

$$f(x^k) \leq f(x^k) + c_k\theta(z^k) \leq \alpha.$$

Thus, the sequence $\{x^k\}$ lies in the level set

$$\{x \in C : f(x) \leq \alpha\},$$

which by assumption is bounded. Thus, (i) holds. To prove (ii), we note that for each $i \notin \mathcal{I}(x^*)$, $\lambda_i^k \leq -g(x^k)$ for all but finitely many $k \in \kappa$; consequently, the sequence $\{\lambda_i^k : k \in \kappa\}$ is bounded. To show that the other components of λ^k and those of μ^k are also bounded, we proceed by contradiction. To simplify the notation somewhat, we assume that $\mathcal{I}(x^*)$ is equal to the full set $\{1, \ldots, m\}$. For each k, we have

$$\|\nabla f(x^k) + \sum_{i=1}^{m} \lambda_i^k \nabla g_i(x^k) + \sum_{j=1}^{q} \mu_j^k \nabla h_j(x^k)\| \leq \theta(z^k).$$

Normalizing by $\|(\lambda^k, \mu^k)\|$, taking limits as $k(\in \kappa) \to \infty$, and noting that $\{\theta(z^k)\}$ is bounded (by part (i) of Lemma 4.2), we deduce that there exists a nonzero pair $(\bar{\lambda}, \bar{\mu})$ with $\bar{\lambda} \geq 0$ satisfying

$$\sum_{i=1}^{m} \bar{\lambda}_i \nabla g_i(x^*) + \sum_{j=1}^{q} \bar{\mu}_j \nabla h_j(x^*) = 0.$$

But this contradicts the two assumed conditions M1 and M2. □

Having provided a sufficient condition for an accumulation point of the sequence $\{z^k\}$ to exist, we next proceed to establish a key property of any accumulation point z^* of this sequence. The following is the main subsequential convergence result for the algorithm described in §3.

THEOREM 4.5. *Let $\{z^k\}$ be an infinite sequence of iterates produced by the algorithm. Then under assumptions* A0–A3, *any accumulation point z^* of $\{z^k\}$ that is* s-regular *is a* KKT *triple of* (1).

Proof. The main idea of the proof is very similar to the proof of the implication (C) ⇒ (A) in Theorem 2.4. By s-regularity, the quadratic program (8) corresponding to z^* has a stationary point $\bar{v}$. Together with some multipliers $\bar{w}$ and $\bar{u}$, $\bar{v}$ satisfies

(27)

$$\nabla f(x^*) + \nabla_{xx}^2 L(z^*)\bar{v} + \sum_{j=1}^{q} \bar{w}_j \nabla h_j(x^*) + \sum_{i \in I(z^*) \cup K_0(z^*)} \bar{u}_i \nabla g_i(x^*) - \sum_{i \in K_+(z^*)} \bar{u}_i \nabla g_i(x^*) = 0,$$

$$\bar{u}_i \geq 0, \quad \bar{u}_i(g_i(x^*) + \nabla g_i(x^*)^T \bar{v}) = 0 \quad \text{for all } i \in K_0(z^*) \cup K_+(z^*).$$

Let $\{z^k : k \in \kappa\}$ be the subsequence whose limit is z^*. For a fixed but arbitrary $k \in \kappa$ and scalar $\tau \in (0,1)$, consider the triple $dz = (dx, d\lambda, d\mu)$ defined by

$$dx = \tau \bar{v}, \qquad d\mu = \tau(\bar{w} - \mu^k),$$

$$d\lambda_i = \begin{cases} \tau(\bar{u}_i - \lambda_i^k) & \text{if } i \in I(z^*) \cup K_0(z^*), \\ -\tau(\bar{u}_i + \lambda_i^k) & \text{if } i \in K_+(z^*), \\ -\tau\lambda_i^k & \text{if } i \in J(z^*) \setminus (K_0(z^*) \cup K_+(z^*)). \end{cases}$$

We also define some error vectors:

$$e^k = \nabla f(x^k) - \nabla f(x^*) + (\nabla^2_{xx} L(z^k) - \nabla^2_{xx} L(z^*))\bar{v} + \sum_{j=1}^{q} \bar{w}_j(\nabla h_j(x^k) - \nabla h_j(x^*)),$$

$$r^k = h(x^k) - h(x^*) + (\nabla h(x^k) - \nabla h(x^*))\bar{v}, \quad s^k = g(x^k) - g(x^*).$$

(We recall that g is an affine function, hence $\nabla g(x)$ is a constant.) The quantities $\|e^k\|, \|r^k\|$, and $\|s^k\|$ all go to zero as $k(\in \kappa) \to \infty$. We claim that for all $\tau > 0$ sufficiently small and all $k \in \kappa$ sufficiently large, the vector dz is feasible to the QP (z^k, B_k) (i.e., $x^k + dx \in C$ and $\lambda^k + d\lambda \geq 0$) and

$$\phi(z^k, dz) \leq (1-\tau)^2 \theta(z^k) + \tau O(\|e^k\| + \|r^k\| + \|s^k\|) + O(\tau^2).$$

Note that $x^k + dx \in C$ is equivalent to $g_i(x^k) + \nabla g_i(x^k)^T dx \leq 0$ for all i. By (27), we have

$$\nabla_x L(z^k) + \nabla^2_{xx} L(z^k) dx + \sum_{i=1}^{m} d\lambda_i \nabla g_i(x^k) + \sum_{j=1}^{q} d\mu_j \nabla h_j(x^k) = (1-\tau)\nabla_x L(z^k) + \tau e^k,$$

which implies

$$\|\nabla_x L(z^k) + \nabla^2_{xx} L(z^k) dx + \sum_{i=1}^{m} d\lambda_i \nabla g_i(x^k) + \sum_{j=1}^{q} d\mu_j \nabla h_j(x^k)\| \leq (1-\tau)\|\nabla_x L(z^k)\| + \tau\|e^k\|.$$

Similarly, we can show

$$h(x^k) + \nabla h(x^k) dx = (1-\tau)h(x^k) + \tau r^k,$$

which yields

$$\|h(x^k) + \nabla h(x^k) dx\| \leq (1-\tau)\|h(x^k)\| + \tau\|r^k\|.$$

Consider an index $i \in I(z^k)$. Then provided that $k \in \kappa$ is sufficiently large, we must have $i \in I(z^*)$ or $\lambda_i^* + g_i(x^*) = 0$. In the former case, we have

$$g_i(x^k) + \nabla g_i(x^k)^T dx = (1-\tau)g_i(x^k) + \tau(g_i(x^k) - g_i(x^*)),$$

which implies

$$(g_i(x^k) + \nabla g_i(x^k)^T dx)^2 \leq (1-\tau)^2 g_i(x^k)^2 + \tau O(\|s^k\|) + O(\tau^2).$$

Moreover, if $g_i(x^*) = 0$, then

$$g_i(x^k) + \nabla g_i(x^k)^T dx = g_i(x^k) \leq 0;$$

on the other hand, if $g_i(x^*) < 0$, then for all $\tau > 0$ sufficiently small and all $k \in \kappa$ sufficiently large, we also have

$$g_i(x^k) + \nabla g_i(x^k)^T dx \leq 0.$$

For the same index i, we have $\lambda_i^* > -g_i(x^*) \geq 0$; consequently, for all $\tau > 0$ sufficiently small and all $k \in \kappa$ sufficiently large, we have

$$\lambda_i^k + d\lambda_i = (1-\tau)\lambda_i^k + \tau \bar{u}_i \geq 0.$$

Thus, the constraints in QP (z^k, B_k) corresponding to this index i are satisfied.

Next, consider an index $i \in I(z^k)$ with $\lambda_i^* + g_i(x^*) = 0$. There are two cases: either $\lambda_i^* = -g_i(x^*) > 0$ (equivalently, $i \in K_+(z^*)$) or $\lambda_i^* = -g_i(x^*) = 0$ (equivalently, $i \in K_0(z^*)$). If $i \in I(z^k) \cap K_+(z^*)$, then

$$\begin{aligned} g_i(x^k) + \nabla g_i(x^k)^T dx &= (1-\tau) g_i(x^k) + \tau(g_i(x^k) - g_i(x^*)) + \tau(g_i(x^*) + \nabla g_i(x^*)^T \bar{v}) \\ &\geq (1-\tau) g_i(x^k) + \tau(g_i(x^k) - g_i(x^*)); \end{aligned}$$

moreover, since $g_i(x^*) < 0$ and $\lambda_i^* > 0$, we have for all $\tau > 0$ sufficiently small and all $k \in \kappa$ sufficiently large,

$$g_i(x^k) + \nabla g_i(x^k)^T dx \leq 0, \quad \lambda_i^k + d\lambda_i = (1-\tau)\lambda_i^k - \tau \bar{u}_i \geq 0.$$

Consequently, the constraints in QP (z^k, B_k) corresponding to this index i are also satisfied, and

$$(g_i(x^k) + \nabla g_i(x^k)^T dx)^2 \leq (1-\tau)^2 g_i(x^k)^2 + \tau O(\|s^k\|) + O(\tau^2).$$

Consider an index $i \in I(z^k) \cap K_0(z^*)$, then

$$g_i(x^k) + \nabla g_i(x^k)^T dx = g_i(x^k) + \tau(g_i(x^*) + \nabla g_i(x^*)^T \bar{v}) \leq 0.$$

Since $g_i(x^*) = 0$, we have

$$\begin{aligned} &(g_i(x^k) + \nabla g_i(x^k)^T dx)^2 = (g_i(x^k) + \tau \nabla g_i(x^*)^T \bar{v})^2 \\ &\leq g_i(x^k)^2 + 2\tau |g_i(x^k)| \, |\nabla g_i(x^*)^T \bar{x}| + O(\tau^2); \end{aligned}$$

moreover, $g_i(x^k)^2$ can be made arbitrarily small for $k \in \kappa$ sufficiently large, in particular, $g_i(x^k)^2 \leq |g_i(x^k)| \leq \|s^k\|$. Hence, we have

$$(g_i(x^k) + \nabla g_i(x^k)^T dx)^2 \leq (1-\tau)^2 g_i(x^k)^2 + \tau O(\|s^k\|) + O(\tau^2)$$

for all $k \in \kappa$ sufficiently large. Furthermore, since $\bar{u}_i \geq 0$ due to $i \in K_0(z^*)$, we have

$$\lambda_i^k + d\lambda_i = (1-\tau)\lambda_i^k + \tau \bar{u}_i \geq 0.$$

Summarizing, we have established that the constraints in QP (z^k, B_k) corresponding to the indices in $I(z^k)$ are all satisfied by the triple dz for all $k \in \kappa$ sufficiently large and all $\tau > 0$ sufficiently small. We still have the remaining constraints corresponding to the indices in $J(z^k)$ to consider. Take an $i \in J(z^k)$. Then we must have $i \in J(z^*)$. There are three cases: (i) $i \in K_0(z^*)$, (ii) $i \in K_+(z^*)$, and (iii) $i \in J(z^*) \setminus (K_0(z^*) \cup K_+(z^*))$. The argument for the first two cases is similar to that given above; in the third case, we have $\lambda_i^* + g_i(x^*) < 0$ which implies $g_i(x^*) < 0$; thus $g_i(x^k) + \nabla g_i(x^k)^T dx \leq 0$ for all $k \in \kappa$ sufficiently large and all $\tau > 0$ sufficiently small. Moreover, we have

$$\lambda_i^k + d\lambda_i = (1 - \tau)\lambda_i^k.$$

Consequently, all the constraints in QP (z^k, B_k) are satisfied. Therefore it follows that

$$(1 - \tau)^2 \theta(z^k) + \tau O(\|e^k\| + \|r^k\| + \|s^k\|) + O(\tau^2) \geq \phi(z^k, dz) + \tfrac{1}{2}(dz)^T B_k dz \geq \gamma_k.$$

Since this must hold for all $k \in \kappa$ sufficiently large, taking the limits as $k(\in \kappa) \to \infty$, by Proposition 4.3, we deduce

$$-2\tau\theta(z^*) + O(\tau^2) \geq 0$$

for all $\tau > 0$ sufficiently small. This yields $\theta(z^*) = 0$ as desired. □

Similar to the other assertions in Theorem 2.4, we obtain the following consequence of Theorem 4.5.

COROLLARY 4.6. *Under assumptions* A0–A3, *if* z^* *is any accumulation point of an infinite sequence* $\{z^k\}$ *produced by the algorithm, and if* (i) $\nabla^2_{xx} L(z^*)$ *is strictly copositive on the cone* $\mathcal{C}(z^*)$, *and* (ii) z^* *satisfies conditions* S1 *and* S2, *then* z^* *is an isolated* KKT *triple of* (1) *and* x^* *is an isolated local minimum of the same program.*

Proof. By Theorem 4.5, we know that z^* is a KKT triple of (1). The rest of the argument is the same as that of Theorem 2.4. □

5. Sequential convergence. Our goal in this section is to show that if the sequence $\{z^k\}$ produced by the algorithm described in §3 has an accumulation point z^* which satisfies two assumptions that are stronger than those in Corollary 4.6, then the entire sequence $\{z^k\}$ will converge to z^* at a Q-quadratic rate. These two assumptions follow.

LICQ (linear independence of active constraint gradients). The gradients

$$\{\nabla h_j(x^*) : j = 1, \ldots, q\} \cup \{\nabla g_i(x^*) : i \in I(z^*) \cup K_0(z^*) \cup K_+(z^*)\}$$

are linearly independent.

SSOSC (strong second-order sufficiency condition). The Hessian matrix $\nabla^2_{xx} L(z^*)$ is positive definite on the linear subspace

$$\mathcal{L}(z^*) \equiv \{v \in R^n : \nabla h_j(x^*)^T v = 0, j = 1, \ldots, q;\ \nabla g_i(x^*)^T v = 0, i \in I(z^*)\}.$$

As noted several times in the preceding discussion, the set $K_+(z^*)$ in LICQ is empty if z^* is a KKT point of (1); in this generalized form, LICQ implies S1 and S2. Moreover, the cone $\mathcal{C}(z^*)$ is clearly a subset of the subspace $\mathcal{L}(z^*)$. Notice that we do not assume any form of strict complementarity on the vector z^*; in other words, we allow the possibility for the set $K_0(z^*)$ to be nonempty.

It follows from assumption A0 that there exists a scalar $\beta > 0$ such that for all vectors $x \in C$ and all vectors $y \in R^n$,

$$y^T \nabla^2 f(x) y \leq \beta \|y\|^2. \tag{28}$$

Fix any scalar $\varepsilon' \in (0,1)$. We assume that the sequence of scalars $\{c_k\}$ satisfies the following condition for all $k \geq 0$ (in addition to assumption A2):

$$\varepsilon' c_{k+1}(\theta(z^k) - \gamma_k) - \nabla f(x^k)^T dx^k - \frac{\beta}{2}\|dx^k\|^2 \geq c_0(\theta(z^k) - \gamma_k), \tag{29}$$

which is somewhat more restrictive than (23).

Variations of the lemma below have been used in [20] and [12] for establishing the rate of convergence of the NE/SQP method and its trust region extension. For this reason, we skip some details in the proof of this lemma and refer the reader to the references where they can be found; see in particular the proofs of Lemmas 8 and 9 in [20]. We recall the function H given by (2) and note that this function is locally Lipschitzian under assumption A0.

LEMMA 5.1. *Suppose z^* is a* KKT *triple of* (1) *satisfying* LICQ *and* SSOSC. *Let $\Lambda > 0$ be the Lipschitzian modulus of the function H at z^*. Then the following statements hold.*

(i) *There exists a neighborhood V of z^* and a constant $\zeta > 0$ such that for all vectors $z \in V \cap Z$ and any symmetric positive definite matrix B,*

$$\|dz\| \leq \zeta \|H(z)\|,$$

where dz is the unique optimal solution of QP (z, B).

(ii) *For any given scalars $\varepsilon, \alpha_2 > 0$, there exists a $\delta > 0$ such that for all vectors $z \in Z$ satisfying $\|z - z^*\| \leq \delta$ and all symmetric positive definite matrices B satisfying $\|B\| \leq \alpha_2 \|H(z)\|$,*

$$\|z + dz - z^*\| \leq \varepsilon \|z - z^*\|, \tag{30}$$

where dz is the unique optimal solution of QP (z, B); *moreover, if $\varepsilon \in (0,1)$ then*

$$\theta(z + dz) \leq \frac{\varepsilon \zeta \Lambda}{1 - \varepsilon} \theta(z).$$

Proof. For an arbitrary vector $z \in Z$, define the $N \times N$ matrix $A(z)$ as follows:

$$A(z) = \begin{bmatrix} \nabla^2_{xx} L(z) & (\nabla g_i(x))_{i \in I(z)} & (\nabla g_i(x))_{i \in J(z)} & \nabla h(x)^T \\ (\nabla g_i(x)^T)_{i \in I(z)} & 0 & 0 & 0 \\ 0 & 0 & I_{J(z)} & 0 \\ \nabla h(x) & 0 & 0 & 0 \end{bmatrix},$$

where $I_{J(z)}$ denotes the identity matrix of order $|J(z)|$. In terms of this matrix, we have

$$\phi(z, dz) = \tfrac{1}{2}\|H(z) + A(z) dz\|^2. \tag{31}$$

It is not difficult to show, by using the LICQ, SSOSC, and the observation that $I(z) \subseteq I(z^*) \cup K_0(z^*)$ for all z sufficiently close to z^*, that the matrix $A(z)$ is nonsingular

for all such vectors z; moreover, there must exist a constant $\alpha_3 > 0$ such that for all vectors z sufficiently close to z^*,

$$\|A(z)^{-1}\| \leq \alpha_3. \tag{32}$$

Since

$$\phi(z, dz) \leq \phi(z, dz) + \tfrac{1}{2} dz^T B dz \leq \phi(z, 0) = \theta(z),$$

the uniform bound on the norm of the inverse of $A(z)$ easily yields the claim (i) with $\zeta \equiv 2\alpha_3$. By the local Lipschitzian property of H, we have

$$\|dz\| \leq \zeta \|H(z) - H(z^*)\| \leq \zeta \Lambda \|z - z^*\|. \tag{33}$$

The expression (30) can be proved as follows. Since the vector $z^* - z$ is feasible to the problem QP (z, B), by the minimum principle for the latter quadratic program, we have

$$0 \leq (z^* - z - dz)^T \left(A(z)^T (H(z) + A(z) dz) + B dz \right),$$

which, after an easy manipulation, yields,

$$\begin{aligned} \|A(z)(z + dz - z^*)\|^2 \leq \|A(z)(z + dz - z^*)\| \big(& \|H(z) - H(z^*) - A(z)(z^* - z)\| \\ & + A(z)^{-T} B dz \| \big). \end{aligned}$$

Canceling one term $\|A(z)(z + dz - z^*)\|$, using the bound (32) for $\|A(z)^{-1}\| = \|A(z)^{-T}\|$ and (33) for $\|dz\|$, and the assumed bound $\|B\| \leq \alpha_2 \|H(z)\|$, we deduce

$$\|z + dz - z^*\| \leq \alpha_3 \left(\|H(z) - H(z^*) - A(z)(z^* - z)\| + \alpha_2 \alpha_3 \Lambda^2 \|z - z^*\|^2 \right).$$

As noted in [12] and [20], we have

$$\lim_{z \to z^*} \frac{H(z) - H(z^*) + A(z)(z^* - z)}{\|z - z^*\|} = 0. \tag{34}$$

Consequently, given $\varepsilon > 0$, we may choose $\delta > 0$ appropriately so that (30) holds. Finally, the last conclusion of part (ii) can be proved by the same argument as in Lemma 9 of [20]. □

Remark. If the second derivatives $\nabla^2 f(x)$ and $\nabla^2 h_j(x)$ are Lipschitzian in a neighborhood of x^*, then we can improve the bound (30) to a quadratic bound; namely, for some constant $\alpha' > 0$,

$$\|z + dz - z^*\| \leq \alpha' \|z - z^*\|^2. \tag{35}$$

This follows because instead of (34), we have

$$\limsup_{z \to z^*} \frac{\|H(z) - H(z^*) + A(z)(z^* - z)\|}{\|z - z^*\|^2} < \infty$$

by the Lipschitzian property of the second derivatives. The improved bound (35) accounts for the quadratic rate of convergence of the sequence $\{z^k\}$.

The following is the main sequential convergence result for the algorithm in §3.

THEOREM 5.2. *Suppose that assumptions* A0–A3 *and condition* (29) *hold. If the sequence* $\{z^k\}$ *has an accumulation point* z^* *which satisfies* LICQ *and* SSOSC, *then*

(i) *the conclusion of Corollary* 4.6 *holds*;

(ii) *there exists an integer* $k_0 > 0$ *such that for all* $k \geq k_0$, *the step size* $\tau_k = \rho^{m_k} = 1$, *hence* $z^{k+1} = z^k + dz^k$;

(iii) *the sequence* $\{z^k\}$ *converges to* z^* Q*-superlinearly, i.e.,*

$$\lim_{k\to\infty} \frac{\|z^{k+1} - z^*\|}{\|z^k - z^*\|} = 0;$$

(iv) *if* $\nabla^2 f$ *and* $\nabla^2 h_j$ *are all Lipschitzian in a neighborhood of* z^*, *then*

$$\limsup_{k\to\infty} \frac{\|z^{k+1} - z^*\|}{\|z^k - z^*\|^2} < \infty.$$

Proof. The first claim follows readily because the assumptions in Corollary 4.6 are all satisfied in the present setting.

Choose a scalar $\varepsilon \in (0,1)$ such that

$$1 - \frac{\varepsilon\zeta\Lambda}{1-\varepsilon} \geq \varepsilon'. \tag{36}$$

Corresponding to this ε, let δ be as specified in part (ii) of Lemma 5.1. Since z^* is an accumulation point of $\{z^k\}$, there exists an integer $k > 0$ such that $\|z^k - z^*\| \leq \delta$. Applying the lemma to this iterate z^k, we deduce

$$\theta(z^k + dz^k) - \theta(z^k) \leq -\left(1 - \frac{\varepsilon\zeta\Lambda}{1-\varepsilon}\right)\theta(z^k). \tag{37}$$

Consequently, for some $\tau \in [0,1]$, we have

$$\begin{aligned}
&P_{c_{k+1}}(z^k + dz^k) - P_{c_{k+1}}(z^k) \\
&= c_{k+1}(\theta(z^k + dz^k) - \theta(z^k)) + f(x^k + dx^k) - f(x^k) \\
&\quad \text{by definition of } P_{c_{k+1}} \\
&\leq -\left(1 - \tfrac{\varepsilon\zeta\Lambda}{1-\varepsilon}\right) c_{k+1}\theta(z^k) + \nabla f(x^k)^T dx^k + \tfrac{1}{2}(dx^k)^T \nabla^2 f(x^k + \tau dx^k) dx^k \\
&\quad \text{by (37) and a mean value theorem} \\
&\leq -\varepsilon' c_{k+1}\theta(z^k) + \nabla f(x^k)^T dx^k + \tfrac{\beta}{2}\|dx^k\|^2 \quad \text{by (28) and (36)} \\
&\leq -c_0(\theta(z^k) - \gamma_k) \quad \text{by (29)} \\
&\leq -\sigma c_0(\theta(z^k) - \gamma_k) \quad \text{since } \sigma \in (0,1) \text{ and } \theta(z^k) \geq \gamma_k.
\end{aligned}$$

This establishes $\tau_k = 1$ for this particular iterate z^k. Thus, $z^{k+1} = z^k + dz^k$ and Lemma 5.1 implies

$$\|z^{k+1} - z^*\| \leq \varepsilon\|z^k - z^*\| < \delta. \tag{38}$$

Consequently, the above derivation can be repeated with z^k replaced by z^{k+1}. Iterating in this manner, we conclude that for some integer $k_0 > 0$, $\tau_k = 1$ and (38) holds for all $k \geq k_0$. Since $\varepsilon > 0$ can be made arbitrarily small, the claims (ii) and (iii) follow. Finally, (iv) follows from the improved inequality (35). □

6. Parallel and inexact implementation. As stated in §3, the algorithm for solving (2) is a serial iterative method whose individual steps must be executed in the prescribed order. Recognizing that each direction-finding subproblem QP (z^k, B_k) is a strictly convex program whose feasible region is naturally "partitionable," we can easily develop variations of the algorithm for implementation on parallel and distributed computers. The objective of this section is to discuss one such implementation.

A natural decomposition. The feasible region Z of QP (z, B) is the Cartesian product of C (the primal polyhedron), R^m_+ (the λ-space), and R^q (the μ-space). To facilitate parallel implementation, we take the matrix B in the objective function to be block diagonal with diagonal blocks B_x, B_λ, and B_μ corresponding to the variables $dx, d\lambda$, and $d\mu$, respectively. Thus, a natural way to solve (5) in parallel is to apply a block coordinate descent method that operates as follows.

Let $z \in Z$ be a fixed vector. Given is an initial triple $dz^0 \in Z - z$ (for instance, $dz^0 = 0$). In general, we are given a triple $dz^\nu \in Z - z$. Solve, in parallel, three subproblems in the variables $dx, d\lambda$, and $d\mu$, respectively; each of these subproblems is obtained from (5) by fixing two of the three groups of variables at the values of the current iterate dz^ν. More specifically, these subproblems are the following.

The dx-subproblem.

$$\begin{aligned} &\text{minimize} && \phi^\nu_x(dx) + \tfrac{1}{2} dx^T B_x dx \\ &\text{subject to} && x + dx \in C, \end{aligned}$$

where $\phi^\nu_x(dx) = \frac{1}{2}(T^\nu_{1,x} + T^\nu_{2,x} + T^\nu_{3,x})$ with

$$\begin{aligned} T^\nu_{1,x} &= \|\nabla_x L(z) + \nabla^2_{xx} L(z) dx + \textstyle\sum_{i=1}^m \nabla g_i(x) d\lambda^\nu_i + \sum_{j=1}^q \nabla h_j(x) d\mu^\nu_j\|^2, \\ T^\nu_{2,x} &= \|h(x) + \nabla h(x) dx\|^2, \quad T^\nu_{3,x} = \textstyle\sum_{i \in I(z)} (g_i(x) + \nabla g_i(x)^T dx)^2; \end{aligned}$$

The $d\lambda$-subproblem.

$$\begin{aligned} &\text{minimize} && \phi^\nu_\lambda(d\lambda) + \tfrac{1}{2} d\lambda^T B_\lambda d\lambda \\ &\text{subject to} && \lambda + d\lambda \geq 0, \end{aligned}$$

where $\phi^\nu_\lambda(d\lambda) = \frac{1}{2}(T^\nu_{1,\lambda} + T^\nu_{4,\lambda})$ with

$$\begin{aligned} T^\nu_{1,\lambda} &= \|\nabla_x L(z) + \nabla^2_{xx} L(z) dx^\nu + \textstyle\sum_{i=1}^m \nabla g_i(x) d\lambda_i + \sum_{j=1}^q \nabla h_j(x) d\mu^\nu_j\|^2, \\ T^\nu_{4,\lambda} &= \textstyle\sum_{i \in J(z)} (\lambda_i + d\lambda_i)^2; \end{aligned}$$

The $d\mu$-subproblem.

$$\text{minimize} \quad \phi^\nu_\mu(d\mu) + \tfrac{1}{2} d\mu^T B_\mu d\mu,$$

where

$$\phi^\nu_\mu(d\mu) = \tfrac{1}{2}\|\nabla_x L(z) + \nabla^2_{xx} L(z) dx^\nu + \textstyle\sum_{i=1}^m \nabla g_i(x) d\lambda^\nu_i + \sum_{j=1}^q \nabla h_j(x) d\mu_j\|^2.$$

Incidentally, this last subproblem is an unconstrained problem in the variable $d\mu$; thus, it is equivalent to a system of linear equations (defined by the symmetric positive definite matrix $B_\mu + \nabla h(x) \nabla h(x)^T$).

Clearly, if the set C in the original nonlinear program (1) has a Cartesian product structure, then this structure can be exploited to refine the above parallel scheme for solving the problem QP (z, B). In general, one could further solve each of the above three subproblems in the variables $dx, d\lambda$, and $d\mu$ by yet another parallel procedure.

Let $dx^{\nu+1/2}, d\lambda^{\nu+1/2}$, and $d\mu^{\nu+1/2}$ be the unique optimal solutions to the above three subproblems, respectively. Write $dz^{\nu+1/2}$ to denote this triple of solutions. Having computed $dz^{\nu+1/2}$, we then perform a synchronization step in which a nonnegative scalar s_ν is determined that solves the following one-dimensional quadratic minimization problem

$$\begin{aligned} &\text{minimize} && \Phi(dz^\nu + s(dz^{\nu+1/2} - dz^\nu)) \\ &\text{subject to} && dz^\nu + s(dz^{\nu+1/2} - dz^\nu) \in Z - z, \quad s \geq 0, \end{aligned}$$

where

$$\Phi(dz) = \phi_z(dz) + \tfrac{1}{2} dz^T B dz.$$

Since Φ is a quadratic function, the determination of s_ν is extremely easy. We then set $dz^{\nu+1} = dz^\nu + s_\nu(dz^{\nu+1/2} - dz^\nu)$. In this manner, a sequence $\{dz^\nu\}$ is generated. The convergence of this sequence to the unique optimal solution of the problem QP (z, B) is known; see, e.g., [7, §5.5]

The above procedure is just one of many possible methods for solving the quadratic programs QP (z, B) in parallel; for other methods, see [1], [9], [16]. Due to the strict convexity of these QPs, one is assured that these parallel methods are all capable of producing an approximate solution to the QPs satisfying any prescribed accuracy in a computationally efficient manner.

Convergence under inexact implementation. Solving each subproblem QP (z^k, B_k) by an infinite scheme (such as the parallel Jacobi method described previously) raises the question of whether the convergence results established in §§4 and 5 will remain valid if the vector dz^k is only an inexactly optimal solution of QP (z^k, B_k). This question is also of practical importance because understanding its answer will facilitate the design of stopping criteria for terminating the "inner" iterations and accepting the inexact iterate dz^k. In [11], an affirmative answer to the question was given for the NE/SQP method and conditions on the inexactness of the approximate solutions to the subproblems were derived that would ensure the validity of the convergence of the method. These conditions can be easily generalized to the present context. The reader is referred to the reference for details.

Acknowledgments. The author is deeply indebted to two referees for their careful reading of the paper and for their many constructive and insightful comments that greatly improved the paper. In particular, one referee has noted that a previous proof of Theorem 5.2 was incorrect due to a different form of assumption A3 used; he suggested the present version of this assumption which leads to the corrected proof.

REFERENCES

[1] D.P. Bertsekas and J.N. Tsitsiklis, *Parallel and Distributed Computation*, Numerical Methods, Prentice-Hall, Englewood Cliffs, NJ, 1989.

[2] J.V. Burke, *A sequential quadratic programming method for potentially infeasible mathematical programs*, J. Math. Anal. Appl., 139 (1989), pp. 319–351.

[3] ———, *An exact penalization view point of constrained optimization*, SIAM J. Control Optim., 29 (1991), pp. 968–998.
[4] ———, *A robust trust region method for constrained nonlinear programming problems*, SIAM J. Optim., 2 (1992), pp. 324–347.
[5] J. V. BURKE AND S. P. HAN, *A robust sequential quadratic programming method*, Math. Programming, 43 (1989), pp. 277–303.
[6] Y. C. CHENG, *Iterative Methods for Linear Complementarity and Programming Problems*, Ph.D. thesis, Computer Science Department, University of Wisconsin, Madison, 1981.
[7] R. W. COTTLE, J. S. PANG, AND R. E. STONE, *The Linear Complementarity Problem*, Academic Press, Boston, 1992.
[8] B. C. EAVES, *On quadratic programming*, Management Sci., 17 (1971), pp. 698–711.
[9] M. C. FERRIS AND O. L. MANGASARIAN, *Parallel constraint distribution*, SIAM J. Optim., 1 (1991), pp. 487–501.
[10] M. C. FERRIS AND S. LUCIDI, *Nonmonotone stabilization methods for nonlinear equations*, J. Optim. Theory Appl., to appear.
[11] S. A. GABRIEL AND J. S. PANG, *An inexact* NE/SQP *method for solving the nonlinear complementarity problem*, Computational Optim. Appl., 1 (1992), pp. 67–92.
[12] ———, *A trust region method for constrained nonsmooth equations*, in Large-Scale Optimization: State of the Art, W. W. Hager, D. W. Hearn, and P. Pardalos, eds., Kluwer Academic Publishers, 1994, Norwell, MA, pp. 159–186.
[13] M. S. GOWDA AND J. S. PANG, *Stability analysis of variational inequalities and nonlinear complementarity problems, via the mixed linear complementarity problem and degree theory*, Math. Oper. Res., 1994, to appear.
[14] S. P. HAN, *A globally convergent method for nonlinear programming*, J. Optim. Theory Appl., 22 (1977), pp. 297–309.
[15] J. KYPARISIS, *On uniqueness of Kuhn–Tucker multipliers in nonlinear programming*, Math. Programming, 32 (1985), pp. 242–246.
[16] Y. Y. LIN AND J. S. PANG, *Iterative methods for large convex quadratic programs: A survey*, SIAM J. Control Optim., 25 (1987), pp. 383–411.
[17] O. L. MANGASARIAN, *Nonlinear Programming*, McGraw-Hill, New York, 1969.
[18] O. L. MANGASARIAN AND S. FROMOVITZ, *The Fritz John necessary optimality conditions in the presence of equality and inequality constraints*, J. Math. Anal. Appl., 17 (1967), pp. 37–47.
[19] J. S. PANG, *A* B*-differentiable equation based, globally, and locally quadratically convergent algorithm for nonlinear programs, complementarity and variational inequality problems*, Math. Programming, 51 (1991), pp. 101–131.
[20] J. S. PANG AND S. A. GABRIEL, NE/SQP: *A robust algorithm for the nonlinear complementarity problem*, Math. Programming, 60 (1993), pp. 295–337.
[21] J. S. PANG AND L. QI, *Nonsmooth equations: motivation and algorithms*, SIAM J. Optim., 3 (1993), pp. 443–465.
[22] M. J. D. POWELL, *A fast algorithm for nonlinearly constrained optimization calculations*, in Proc. 1977 Dundee Biennial Conference on Numerical Analysis, G.A. Watson, ed., Springer-Verlag, Berlin, 1978, pp. 144–157.
[23] S. M. ROBINSON, *Local structure of feasible sets in nonlinear programming, Part* III*: Stability and sensitivity*, Math. Programming Study, 30 (1987), pp. 45–66.
[24] R. T. ROCKAFELLAR, *Convex Analysis*, Princeton University Press, Princeton, NJ, 1970.
[25] P. K. SUBRAMANIAN, *Iterative Methods of Solutions for Complementarity Problems*, Ph.D. thesis, Computer Sciences Department, University of Wisconsin, Madison, 1985.

SIAM J. OPTIMIZATION
Vol. 4, No. 4, pp. 894–904, November 1994
© 1994 Society for Industrial and Applied Mathematics
012

A STOCHASTIC METHOD FOR CONSTRAINED GLOBAL OPTIMIZATION*

KLAUS RITTER† AND STEFAN SCHÄFFLER†

Abstract. A stochastic method is described for solving constrained global optimization problems using a penalty approach. Interesting properties and a parallel implementation of the algorithm enable the treatment of problems with a large number of variables.

Key words. constrained global optimization, penalty approach, Brownian motion, stochastic method, parallel algorithms, transputer networks

AMS subject classifications. 49M37, 60H20, 90C30

1. Introduction.

The following constrained global optimization problem is considered:

$$\text{(CGOP)} \qquad \begin{aligned} &\min_{\mathbf{x}} \{F(\mathbf{x}) \mid h_i(\mathbf{x}) \leq 0 \quad i = 1, \ldots, m \\ &\qquad\qquad h_i(\mathbf{x}) = 0 \quad i = m+1, \ldots, m+q\}, \\ &F, h_i : \mathbb{R}^n \to \mathbb{R};\ F,\ h_i \in C^2, \qquad i = 1, \ldots, m+q. \end{aligned}$$

During recent years, many papers concerning global optimization have been published (see the summaries in [4], [5], and [18]). In this paper a penalty approach for solving (CGOP) is used. Under appropriate assumptions, the sequence of global minimizers of the unconstrained subproblems converges to a global minimizer of (CGOP). For the solutions of the unconstrained global optimization problems, a new stochastic method of Schäffler [16] is applied that is implemented in parallel on a transputer network. Section 2 deals with preliminaries. In §3 the results for unconstrained global optimization given in [16] are summarized. Constrained global optimization and the formulation of an algorithm and its properties are contained in §4. Finally, in §5 a parallel implementation on a transputer network is given and numerical results are considered.

2. Preliminaries.

Let Ω^n be the set of all continuous functions $\mathbf{v}$: $[0, \infty[\to \mathbb{R}^n$, $n \in \mathbb{N}$. We define a metric d on Ω^n by

$$d(\omega_1^n, \omega_2^n) := \sum_{m=1}^{\infty} \frac{1}{2^m} \min\left(\max_{0 \leq t \leq m} \|\omega_1^n(t) - \omega_2^n(t)\|_2;\ 1\right),$$

where $\|\cdot\|_2$ denotes the Euclidean norm. By $\mathcal{B}(\Omega^n)$ we mean the smallest σ-field containing all open sets of Ω^n in the topology defined by the metric d. The open sets in $\mathbb{R}^p$, $p \in \mathbb{N}$ are given by the Euclidean norm. Throughout this paper we use only the probability spaces $(\Omega^n, \mathcal{B}(\Omega^n), W^n)$, $n \in \mathbb{N}$, where W^n denotes an n-dimensional Wiener measure that we consider later. Let $\bar{\mathbb{R}} := \mathbb{R} \cup \{\pm\infty\}$ be the compactification of $\mathbb{R}$ with $0 \cdot (\infty) = 0 \cdot (-\infty) = (\infty) \cdot 0 = (-\infty) \cdot 0 := 0$ and let $\mathcal{B}(\bar{\mathbb{R}})$ be the Borel σ-field of $\bar{\mathbb{R}}$ given by

$$B \in \mathcal{B}(\bar{\mathbb{R}}) \Leftrightarrow (B \cap \mathbb{R}) \in \mathcal{B}(\mathbb{R}).$$

Now we summarize some important definitions (see [1] and [15]).

DEFINITION 2.1. *A function $g : \Omega^n \to \bar{\mathbb{R}}$ with $n \in \mathbb{N}$ is called a numerical function.*

DEFINITION 2.2. *Let $p \in \mathbb{N}$. A collection $\{\mathbf{X}_t\}$ of p-dimensional random variables $\mathbf{X}_t$, $t \geq 0$, defined on Ω^p is called a stochastic process.*

* Received by the editors September 7, 1993; accepted for publication (in revised form) February 8, 1994.
† Institut für Angewandte Mathematik und Statistik, Technische Univerität München, Arcisstr. 21, Postfach, D-80290 München, Germany (ritter@statistik.tu-muenchen.de).

DEFINITION 2.3. *Let* $\{\mathbf{X}_t\}$ *be a* p-dim *stochastic process. For each fixed* $\omega^p \in \Omega^p$, *the function* $\mathbf{X}_{\omega^p} : [0, \infty[\to \mathbb{R}^p$, $t \mapsto \mathbf{X}_t(\omega^p)$ *is called a path of* $\{\mathbf{X}_t\}$.

DEFINITION 2.4. *Let* $\{\mathbf{X}_t\}$ *be a stochastic process.* $\{\mathbf{X}_t\}$ *is called continuous if each path of* $\{\mathbf{X}_t\}$ *is a continuous function.*

For each $n \in \mathbb{N}$ our Wiener measure W^n is uniquely determined by the following conditions using the stochastic process $\{\mathbf{B}_t\}$ defined by $\mathbf{B}_t(\omega^n) := \omega^n(t)$ for all $t \geq 0$ (a proof is given in [1]).

(i) $\mathbf{B}_0(\omega^n) = \mathbf{0}$ W^n almost surely.

(ii) For every $0 = t_0 < t_1 < \cdots < t_m$, $m \in \mathbb{N}$, the random variables $\mathbf{B}_0$, $\mathbf{B}_{t_1} - \mathbf{B}_{t_0}, \ldots, \mathbf{B}_{t_m} - \mathbf{B}_{t_{m-1}}$ are independent.

(iii) For every $0 \leq s < t$ the random variable $\mathbf{B}_t - \mathbf{B}_s$ is $N(\mathbf{0}, (t-s)\mathbf{I}_n)$ Gaussian distributed, where $\mathbf{I}_n$ denotes the n-dim identity matrix.

Now we are able to define the Brownian motion process.

DEFINITION 2.5. *The stochastic process* $\{\mathbf{B}_t\}$ *defined by* $\mathbf{B}_t(\omega^n) := \omega^n(t)$ *is called* n*-dimensional Brownian motion.*

The following definition is very important for the investigation of some properties of stochastic processes.

DEFINITION 2.6. *Let* $n \in \mathbb{N}$. *A* $\mathcal{B}(\Omega^n) - \mathcal{B}(\bar{\mathbb{R}})$ *measurable numerical function* g *is called a random time.*

3. The unconstrained case. Now we consider a new stochastic method for the solution of the unconstrained global optimization problem.

(UGOP) Compute $\mathbf{x}^* \in \mathbb{R}^n$ with $f(\mathbf{x}^*) \leq f(\mathbf{x})$ for all $\mathbf{x} \in \mathbb{R}^n$,
where $f : \mathbb{R}^n \to \mathbb{R}$, $f \in C^2$ is a given objective function.

The results of this section are restated from [16] without proofs. The following assumption concerning the objective function is fundamental for further investigations.

Assumption A. There exists an $\varepsilon > 0$ such that

$$\mathbf{x}^{\mathrm{T}} \nabla f(\mathbf{x}) \geq \frac{1 + n\varepsilon^2}{2} \max(1, \|\nabla f(\mathbf{x})\|_2) \qquad \text{for all } \mathbf{x} \in \mathbb{R}^n \backslash \{\mathbf{x} \in \mathbb{R}^n \mid \|\mathbf{x}\|_2 \leq r\}$$

with some $r > 0$, where $\nabla f(\mathbf{x})$ denotes the gradient of f at $\mathbf{x}$.

Assumption A describes a special behaviour of f outside a ball with radius r for which only the existence is postulated. Starting at the origin, the objective function f must increase sufficiently fast along each direction $\mathbf{s} \in \mathbb{R}^n$, $\|\mathbf{s}\|_2 = 1$ outside the mentioned ball. It is obvious that each function for which Assumption A is fulfilled has a global minimizer. The opposite is not true as the function $\mathbf{x} \mapsto \sin(\mathbf{x})$ shows.

For the solution of (UGOP) we investigate the following class of stochastic integral equations (SIE), where $\{\mathbf{B}_t\}$ is an n-dim Brownian motion, $\omega^n \in \Omega^n$, and f is the given objective function.

$$(\mathrm{SIE})_{\varepsilon,\mathbf{x}_0} \qquad \mathbf{X}_t^{\varepsilon,\mathbf{x}_0}(\omega^n) = \mathbf{x}_0 + \varepsilon(\mathbf{B}_t(\omega^n) - \mathbf{B}_0(\omega^n)) - \int_0^t \nabla f(\mathbf{X}_\tau^{\varepsilon,\mathbf{x}_0}(\omega^n))d\tau,$$

with $\varepsilon > 0$, $\mathbf{x}_0 \in \mathbb{R}^n$. Now we study the existence, uniqueness, and regularity of the solutions of (SIE).

THEOREM 3.1. *Consider* (SIE). *For all* $\mathbf{x}_0 \in \mathbb{R}^n$ *and for all* ε, *for which Assumption* A *is fulfilled, we obtain the following:*

(a) *there exists a unique stochastic process* $\{\mathbf{X}_t^{\varepsilon,\mathbf{x}_0}\}$ *that solves* $(\mathrm{SIE})_{\varepsilon,\mathbf{x}_0}$;

(b) *all paths of* $\{\mathbf{X}_t^{\varepsilon,\mathbf{x}_0}\}$ *are continuous*;
(c) $\mathbf{X}_0^{\varepsilon,\mathbf{x}_0} \equiv \mathbf{x}_0$.

Let $\bar{\mathbf{x}} \in \mathbb{R}^n$ and consider $S(\bar{\mathbf{x}}, \rho) := \{\mathbf{x} \in \mathbb{R}^n \mid \|\mathbf{x} - \bar{\mathbf{x}}\|_2 \le \rho\}$, $\rho > 0$. For the investigation of the relations between the solutions of (SIE) and (UGOP), we need the random times $s_{\bar{\mathbf{x}},\rho} : \Omega^n \to \bar{\mathbb{R}}$,

$$\omega^n \mapsto \begin{cases} \inf\{t \ge 0 \mid \|\mathbf{X}_t^{\varepsilon,\mathbf{x}_0}(\omega^n) - \bar{\mathbf{x}}\|_2 \le \rho\} & \text{if } \{t \ge 0 \mid \|\mathbf{X}_t^{\varepsilon,\mathbf{x}_0}(\omega^n) - \bar{\mathbf{x}}\|_2 \le \rho\} \ne \emptyset, \\ \infty & \text{else.} \end{cases}$$

Theorem 3.1 allows us to apply some important results from [10] that we formulate in Theorem 3.2. The proof of this theorem leads to Lyapunov's stability calculus for stochastic differential equations and to the theory of parabolic partial differential equations.

THEOREM 3.2. *Consider* (SIE) *with* $\varepsilon > 0$ *such that Assumption* A *is fulfilled. Then we obtain the following for each* $\mathbf{x}_0, \bar{\mathbf{x}} \in \mathbb{R}^n$, $\rho > 0$.

(a) $W^n(\{\omega^n \in \Omega^n \mid s_{\bar{\mathbf{x}},\rho}(\omega^n) < \infty\}) = 1$.
(b) $E(s_{\bar{\mathbf{x}},\rho}) < \infty$ ($E(.)$ *denotes expectation*).
(c) *Each random variable* $\mathbf{X}_t^{\varepsilon,\mathbf{x}_0}$, $t > 0$, *has a density* $p_{\varepsilon,\mathbf{x}_0,t}$ *with*

$$\lim_{t\to\infty} p_{\varepsilon,\mathbf{x}_0,t}(\mathbf{x}) = p_\varepsilon(\mathbf{x}),$$

where p_ε *is the unique density function satisfying*

$$\frac{1}{2}\varepsilon^2 \sum_{i=1}^n \frac{\partial^2 p_\varepsilon(\mathbf{x})}{\partial x_i^2} + \sum_{i=1}^n \frac{\partial}{\partial x_i}\left(\frac{\partial f(\mathbf{x})}{\partial x_i} p_\varepsilon(\mathbf{x})\right) \equiv 0.$$

It is important to recognize that the limit p_ε is independent of the starting point $\mathbf{x}_0$. Now we are interested in the density p_ε and its relationship to (UGOP). For the explicit formulation of p_ε we need the following theorem.

THEOREM 3.3. *Let* f_{gl} *be the global minimum of* f, *then*

$$C_\varepsilon := \int_{\mathbb{R}^n} \exp\left(-\frac{2(f(\mathbf{x}) - f_{\mathrm{gl}})}{\varepsilon^2}\right) d\mathbf{x} < \infty$$

for all ε *satisfying Assumption* A.

With Theorem 3.3 we are able to give p_ε explicitly:

$$p_\varepsilon(\mathbf{x}) = \frac{1}{C_\varepsilon} \exp\left(-\frac{2(f(\mathbf{x}) - f_{\mathrm{gl}})}{\varepsilon^2}\right)$$

Each global minimizer of f is a global maximizer of p_ε. This is the most important relationship between the solutions of (SIE), where the densities of $\mathbf{X}_t^{\varepsilon,\mathbf{x}_0}$ converge pointwise to p_ε, and (UGOP). From Theorem 3.2 we know that for each local minimizer $\bar{\mathbf{x}}$ W^n almost all paths of $\{\mathbf{X}_t^{\varepsilon,\mathbf{x}_0}\}$ hit any ball $S(\bar{\mathbf{x}}, \xi)$ centered at $\bar{\mathbf{x}}$ for arbitrary chosen $\xi > 0$ after a finite time for all $\mathbf{x}_0 \in \mathbb{R}^n$. Moreover, the expectation of $s_{\bar{\mathbf{x}},\rho}$ is finite.

Now we study a semi-implicit Euler method for the numerical solution of (SIE) at a fixed but arbitrary chosen $\omega^n \in \Omega^n$. The solution $\mathbf{X}_{t+\sigma}^{\varepsilon,\xi}(\omega^n)$ of

$$\mathbf{X}_{t+\sigma}^{\varepsilon,\xi}(\omega^n) = \xi + \varepsilon(\mathbf{B}_{t+\sigma}(\omega^n) - \mathbf{B}_t(\omega^n)) - \int_t^{t+\sigma} \nabla f(\mathbf{X}_\tau^{\varepsilon,\xi}(\omega^n))d\tau$$

is denoted by $\mathbf{X}^\varepsilon(t+\sigma; \xi, \sigma, \omega^n)$, $\sigma \ge 0$, $\xi \in \mathbb{R}^n$. Using the Taylor expansion

$$\nabla f(\mathbf{x}) = \nabla f(\xi) + \nabla^2 f(\xi)(\mathbf{x} - \xi) + \int_0^1 (\nabla^2 f(\xi + \vartheta(\mathbf{x} - \xi)) - \nabla^2 f(\xi))d\vartheta(\mathbf{x} - \xi),$$

we obtain the following approximation $\bar{\mathbf{X}}^{\varepsilon}(t+\sigma;\,\xi,\sigma,\omega^n)$ for $\mathbf{X}^{\varepsilon}(t+\sigma;\,\xi,\sigma,\omega^n)$:

$$\bar{\mathbf{X}}^{\varepsilon}(t+\sigma;\,\xi,\sigma,\omega^n) = \xi + \varepsilon(\mathbf{B}_{t+\sigma}(\omega^n) - \mathbf{B}_t(\omega^n)) - \sigma\nabla f(\xi) - \sigma\nabla^2 f(\xi)(\bar{\mathbf{X}}^{\varepsilon}(t+\sigma;\,\xi,\sigma,\omega^n) - \xi),$$

and thus

$$\bar{\mathbf{X}}^{\varepsilon}(t+\sigma;\,\xi,\sigma,\omega^n) = \xi - (\mathbf{I}_n + \sigma\nabla^2 f(\xi))^{-1}(\sigma\nabla f(\xi) - \varepsilon(\mathbf{B}_{t+\sigma}(\omega^n) - \mathbf{B}_t(\omega^n)))$$

for all σ with $(\mathbf{I}_n + \sigma\nabla^2 f(\xi))$ regular.

Since $\mathbf{B}_{t+\sigma} - \mathbf{B}_t$ has an $N(\mathbf{0}, \sigma\mathbf{I}_n)$ Gaussian distribution, we compute $\mathbf{B}_{t+\sigma}(\omega^n) - \mathbf{B}_t(\omega^n)$ by stimulation of an $N(\mathbf{0}, \mathbf{I}_n)$ Gaussian-distributed random vector. For the integration error $\|\mathbf{X}^{\varepsilon}(t+\sigma;\,\xi,\sigma,\omega^n) - \bar{\mathbf{X}}^{\varepsilon}(t+\sigma;\,\xi,\sigma,\omega^n)\|_2$, we obtain the following result.

THEOREM 3.4. *For each ε such that Assumption* A *is fulfilled and for each $t \geq 0$, $\xi \in \mathbb{R}^n$, we obtain the existence of $c > 0$ with*

$$\lim_{\sigma\to 0} \frac{\|\mathbf{X}^{\varepsilon}(t+\sigma;\,\xi,\sigma,\omega^n) - \bar{\mathbf{X}}^{\varepsilon}(t+\sigma;\,\xi,\sigma,\omega^n)\|_2}{\sqrt{\sigma^3 \ln(\ln(1/\sigma))}} \leq c \qquad W^n \text{ almost surely.}$$

For stepsize σ, only such σ with $(\mathbf{I}_n + \sigma\nabla^2 f(\xi))$ positive definite are used. Beginning with a starting value σ^*, we compute $\sigma \leq \sigma^*$ such that $(\mathbf{I}_n + \sigma\nabla^2 f(\xi))$ is positive definite and we compute

$$\bar{\mathbf{X}}^{\varepsilon}\left(t+\frac{\sigma}{2};\,\xi,\frac{\sigma}{2},\omega^n\right), \quad \bar{\mathbf{X}}^{\varepsilon}(t+\sigma;\,\xi,\sigma,\omega^n), \quad \text{and}$$

$$\bar{\mathbf{X}}^{\varepsilon}\left(t+\sigma;\,\bar{\mathbf{X}}^{\varepsilon}\left(t+\frac{\sigma}{2};\,\xi,\frac{\sigma}{2},\omega^n\right),\frac{\sigma}{2},\omega^n\right).$$

Choose $\delta > 0$. If

$$\left\| \bar{\mathbf{X}}^{\varepsilon}(t+\sigma;\,\xi,\sigma,\omega^n) - \bar{\mathbf{X}}^{\varepsilon}\left(t+\sigma;\,\bar{\mathbf{X}}^{\varepsilon}\left(t+\frac{\sigma}{2};\,\xi,\frac{\sigma}{2},\omega^n\right),\frac{\sigma}{2},\omega^n\right) \right\|_2 < \delta,$$

then σ is accepted that

$$\bar{\mathbf{X}}^{\varepsilon}\left(t+\sigma;\,\bar{\mathbf{X}}^{\varepsilon}\left(t+\frac{\sigma}{2};\,\xi,\frac{\sigma}{2},\omega^n\right),\frac{\sigma}{2},\omega^n\right)$$

is the approximation for $\mathbf{X}^{\varepsilon}(t+\sigma;\xi,\sigma,\omega^n)$, else choose $\sigma/2$ as a new stepsize.

It is important to recognize that the realizations

$$\mathbf{s}_1 := \varepsilon(\mathbf{B}_{t+\sigma/2}(\omega^n) - \mathbf{B}_t(\omega^n)) \quad \text{and} \quad \mathbf{s}_2 := \varepsilon(\mathbf{B}_{t+\sigma}(\omega^n) - \mathbf{B}_{t+\sigma/2}(\omega^n)),$$

which are used for the computation of

$$\bar{\mathbf{X}}^{\varepsilon}\left(t+\frac{\sigma}{2};\,\xi,\frac{\sigma}{2},\omega^n\right) \quad \text{and} \quad \bar{\mathbf{X}}^{\varepsilon}\left(t+\sigma;\,\bar{\mathbf{X}}^{\varepsilon}\left(t+\frac{\sigma}{2};\,\xi,\frac{\sigma}{2},\omega^n\right),\frac{\sigma}{2},\omega^n\right),$$

must be used for the computation of $\bar{\mathbf{X}}^{\varepsilon}(t+\sigma;\,\xi,\sigma,\omega^n)$, because

$$\bar{\mathbf{X}}^{\varepsilon}\left(t+\sigma;\,\bar{\mathbf{X}}^{\varepsilon}\left(t+\frac{\sigma}{2};\,\xi,\frac{\sigma}{2},\omega^n\right),\frac{\sigma}{2},\omega^n\right), \quad \bar{\mathbf{X}}^{\varepsilon}\left(t+\frac{\sigma}{2};\,\xi,\frac{\sigma}{2},\omega^n\right), \quad \text{and}$$

$$\bar{\mathbf{X}}^{\varepsilon}(t+\sigma;\,\xi,\sigma,\omega^n)$$

are approximations of points of the same path. At the end of this section we investigate the question of how to choose ε. For that, we consider (SIE):

$$\mathbf{X}_t^{\varepsilon,\mathbf{x}_0}(\omega^n) = \mathbf{x}_0 + \varepsilon(\mathbf{B}_t(\omega^n) - \mathbf{B}_0(\omega^n)) - \int_0^t \nabla f(\mathbf{X}_\tau^{\varepsilon,\mathbf{x}_0}(\omega^n))d\tau.$$

The parameter ε is a measure for the balance between local minimization along the curve of steepest descent

$$\mathbf{X}_t^{\mathbf{x}_0} = \mathbf{x}_0 - \int_0^t \nabla f(\mathbf{X}_\tau^{\mathbf{x}_0})d\tau,$$

and a random search with realizations of Gaussian-distributed random vectors with increasing variance

$$\mathbf{X}_t^{\mathbf{x}_0}(\omega^n) = \mathbf{x}_0 + \mathbf{B}_t(\omega^n) - \mathbf{B}_0(\omega^n).$$

If we choose ε for a fixed ω^n such that local minimization dominates, then the chosen path of $\{\mathbf{X}_t^{\varepsilon,\mathbf{x}_0}\}$ spends a very long time close to (local) minimizers of f. If we choose ε such that random search dominates, then the minimizers of f play no significant role along this path of $\{\mathbf{X}_t^{\varepsilon,\mathbf{x}_0}\}$. The optimal balance between random search and local minimization, and therefore the choice of ε, depends on the objective function f and the scale used. If one observes, during the numerical computations of a path of $\{\mathbf{X}_t^{\varepsilon,\mathbf{x}_0}\}$, that this path spends a very long time close to local minimizers of f, then ε is too small. If the minimizers of f play no significant role along this path, then ε is too large.

Now we discuss the use of cooling strategies (functions $\varepsilon : \mathbb{R} \to \mathbb{R}$). Consider the density

$$p_\varepsilon(\mathbf{x}) = \frac{1}{C_\varepsilon} \exp\left(-\frac{2(f(\mathbf{x}) - f_{\mathrm{gl}})}{\varepsilon^2}\right).$$

For decreasing $\varepsilon > 0$, the probability measure given by p_ε is more concentrated on the global minimizers of f. This is the idea of cooling strategies: Find a strictly monotone-decreasing function $\varepsilon : \mathbb{R} \to \mathbb{R}$ that converges to 0, for which the probability measures given by p_ε are more and more concentrated on the global minimizers of f. Theoretical investigations (see [9]) show that under special assumptions on the objective function f (comparable to our Assumption A) cooling strategies must be of the following types:

(i) $\varepsilon(0) \geq C$, where C is an unknown constant,
(ii) $\lim_{t\to\infty} \varepsilon(t) = 0$,
(iii) $\lim_{t\to\infty} \varepsilon^2(t)\ln(t) > 0$.

If $\varepsilon(0) < C$ or $\lim_{t\to\infty} \varepsilon^2(t)\ln(t) = 0$, the probability measure given by the limit distribution is only concentrated on local minimizers. The convergence to the limit distribution is very slow as (iii) shows. Therefore, no numerical results using cooling strategies are published.

4. Constrained global optimization. We consider the following constrained global optimization problem.

$$\text{(CGOP)} \qquad \begin{aligned} \min_{\mathbf{x}} \{F(\mathbf{x}) \mid h_i(\mathbf{x}) &\leq 0 \quad i = 1,\dots,m \\ h_i(\mathbf{x}) &= 0 \quad i = m+1,\dots,m+q\}, \\ F, h_i : \mathbb{R}^n \to \mathbb{R};\ F, h_i &\in C^2, \qquad i = 1,\dots,m+q. \end{aligned}$$

Using

$$P : \mathbb{R} \to \mathbb{R},\ x \mapsto \begin{cases} x & \text{if } x \geq 0, \\ 0 & \text{else,} \end{cases}$$

we define the following function:

$$p: \mathbb{R}^n \to \mathbb{R}, \quad \mathbf{x} \mapsto F(\mathbf{x}) + \mu \left(\sum_{i=1}^{m} P(h_i(\mathbf{x}))^4 + \sum_{i=m+1}^{m+q} h_i(\mathbf{x})^2 \right), \quad \mu \in \mathbb{R}^+.$$

The fourth power for inequalities is chosen to ensure that $p \in C^2$.

Now we introduce our main assumption.

Assumption B. There exists $\mu_0 \in \mathbb{R}^+$ such that for all $\mu \geq \mu_0$ there is an

$$\varepsilon > 0 \text{ with } \mathbf{x}^T \nabla p(\mathbf{x}) \geq \frac{1 + n\varepsilon^2}{2} \max(1, \|\nabla p(\mathbf{x})\|_2)$$
$$\text{for all } \mathbf{x} \in \mathbb{R}^n \backslash \{\mathbf{x} \in \mathbb{R}^n \mid \|\mathbf{x}\|_2 \leq r\} \text{ with some } r > 0.$$

Assumption B means that Assumption A of the unconstrained case is fulfilled for all functions p with $\mu \geq \mu_0$. It is obvious that each function p with $\mu \geq \mu_0$ has a global minimizer if Assumption B is satisfied.

Consider sequences $\{\mu_j\}$, $j \in \mathbb{N}$ of penalty parameters with

(i) $\mu_{j+1} > \mu_j > \mu_0$ for all $j \in \mathbb{N}$, and

(ii) $\lim_{j \to \infty} \mu_j = \infty$.

For each j we define

$$p_j: \mathbb{R}^n \to \mathbb{R}, \mathbf{x} \mapsto F(\mathbf{x}) + \mu_j \left(\sum_{i=1}^{m} P(h_i(\mathbf{x}))^4 + \sum_{i=m+1}^{m+q} h_i(\mathbf{x})^2 \right).$$

A global minimizer of p_j is denoted by $\mathbf{x}_j^*$.

THEOREM 4.1. *Consider* (CGOP) *and let Assumption* B *be fulfilled. Then it holds that*

(a) (CGOP) *has a solution*;

(b) *each cluster point of any sequence* $\{\mathbf{x}_j^*\}$, $j \in \mathbb{N}$ *is a solution of* (CGOP).

Proof. Let $R := \{\mathbf{x} \in \mathbb{R}^n \mid h_i(\mathbf{x}) \leq 0, i = 1, \ldots, m;\ h_i(\mathbf{x}) = 0, i = m+1, \ldots, m+q\}$ be the feasible region of (CGOP) and define $F_{\inf} := \inf\{F(\mathbf{x}) \mid \mathbf{x} \in R\}$. From the definition of p_j and $\mathbf{x}_j^*$, we obtain

$$p_j(\mathbf{x}_j^*) \leq F_{\inf} \quad \text{for all } j \in \mathbb{N}. \tag{4.1}$$

Consider any sequence $\{\mathbf{x}_j^*\}$, $j \in \mathbb{N}$. Assumption B guarantees that $\mathbf{x}_j^* \in \{\mathbf{x} \in \mathbb{R}^n \mid \|\mathbf{x}\|_2 \leq r\}$ for all $j \in \mathbb{N}$. Hence, each sequence $\{\mathbf{x}_j^*\}$ has cluster points. Let $\mathbf{x}_c^*$ be a cluster point of $\{\mathbf{x}_j^*\}$ and let $\{\mathbf{x}_k^*\}$ be a subsequence of $\{\mathbf{x}_j^*\}$ that converges to $\mathbf{x}_c^*$. Assume $\mathbf{x}_c^* \notin R$; then we get from the definition of $\{\mu_j\}$ and from the fact that R is closed that

$$\lim_{k \to \infty} p_k(\mathbf{x}_k^*) = \infty,$$

which is a contradiction to (4.1). Assume that $F(\mathbf{x}_c^*) > F_{\inf}$, then $p_j(\mathbf{x}_c^*) > F_{\inf}$ for all $j \in \mathbb{N}$. This is a contradiction to (4.1) and the convergence of $\{\mathbf{x}_k^*\}$. Hence, $F(\mathbf{x}_c^*) = F_{\inf}$. □

Since efficient local minimization procedures exist for constrained optimization (see, e.g., [17]), we are only interested in a neighborhood of a solution of (CGOP). Therefore, only one unconstrained global optimization problem with objective function p and penalty parameter $\mu = 1000$ is solved. For that, the following algorithm, which is based on the semi-implicit Euler method of §3, is used.

Step 0 (Initialization).

Choose $\mathbf{x}_0$, ε, δ,

$j := 0$,

goto Step 1.

In Step 0, *the starting point* $\mathbf{x}_0$, *the parameter* ε (*cf. Assumption* B), *and the parameter* δ (*for the stepsize control*) *must be determined by the user.*

Step 1 (Derivatives).

$\sigma := 1$,

compute $\nabla p(\mathbf{x}_j)$, $\nabla^2 p(\mathbf{x}_j)$,

goto Step 2.

The initial value for the stepsize is equal to 1. *The derivatives are computed using automatic differentiation.*

Step 2 (Simulation).

Compute realizations $\mathbf{n}_1$ and $\mathbf{n}_2$ of two independent, $N(\mathbf{0}, \mathbf{I}_n)$.

Gaussian distributed random vectors,

goto Step 3.

In this step, the path of $\{\mathbf{X}_t^{\varepsilon,\mathbf{x}_0}\}$ *for the numerical calculations is determined successively.*

Step 3 (Cholesky).

Compute $\mathbf{L}$ with

$\mathbf{L}\mathbf{L}^{\mathrm{T}} = \mathbf{I}_n + \sigma\nabla^2 p(\mathbf{x}_j)$.

Case 1. $\mathbf{I}_n + \sigma\nabla^2 p(\mathbf{x}_j)$ is not positive definite.

$\sigma := 0.5\sigma$

goto Step 3.

Case 2. $\mathbf{I}_n + \sigma\nabla^2 p(\mathbf{x}_j)$ is positive definite.

goto Step 4.

It is obvious to use the Cholesky factorization of $\mathbf{I}_n + \sigma\nabla^2 p(\mathbf{x}_j)$ *for the computation of* $\bar{\mathbf{X}}^{\varepsilon}(\sigma; \mathbf{x}_j, \sigma, \omega^n)$.

Step 4 (Computation of $\mathbf{x}_{j+1}(\sigma)$).

Compute $\mathbf{x}_{j+1}(\sigma)$ via

$\mathbf{L}\mathbf{L}^{\mathrm{T}}\mathbf{x}_{j+1}(\sigma) = \sigma\nabla p(\mathbf{x}_j) - \varepsilon\sqrt{\sigma/2}\,(\mathbf{n}_1 + \mathbf{n}_2)$ and

$\mathbf{x}_{j+1}(\sigma) := \mathbf{x}_j - \mathbf{x}_{j+1}(\sigma)$,

goto Step 5.

$\bar{\mathbf{X}}^{\varepsilon}(\sigma; \mathbf{x}_j, \sigma, \omega^n)$ *is denoted by* $\mathbf{x}_{j+1}(\sigma)$.

Step 5 (Cholesky).

Compute $\mathbf{L}$ with

$\mathbf{L}\mathbf{L}^{\mathrm{T}} = \mathbf{I}_n + \frac{\sigma}{2}\nabla^2 p(\mathbf{x}_j)$,

goto Step 6.

The matrix $\mathbf{I}_n + \frac{\sigma}{2}\nabla^2 p(\mathbf{x}_j)$ *is obviously positive definite.*

Step 6 (Computation of $\mathbf{x}_{j+1}(\frac{\sigma}{2})$).

Compute $\mathbf{x}_{j+1}(\frac{\sigma}{2})$ via

$\mathbf{L}\mathbf{L}^{\mathrm{T}}\mathbf{x}_{j+1}(\frac{\sigma}{2}) = \frac{\sigma}{2}\nabla p(\mathbf{x}_j) - \varepsilon\sqrt{\sigma/2}\,\mathbf{n}_1$ and

$\mathbf{x}_{j+1}(\frac{\sigma}{2}) := \mathbf{x}_j - \mathbf{x}_{j+1}(\frac{\sigma}{2})$,

goto Step 7.

$\bar{\mathbf{X}}^{\varepsilon}(\frac{\sigma}{2}; \mathbf{x}_j, \frac{\sigma}{2}, \omega^n)$ *is denoted by* $\mathbf{x}_{j+1}(\frac{\sigma}{2})$.

Step 7 (Derivatives).

Compute $\nabla p(\mathbf{x}_{j+1}(\frac{\sigma}{2}))$, $\nabla^2 p(\mathbf{x}_{j+1}(\frac{\sigma}{2}))$,

goto Step 8.

Step 8 (Cholesky).

Compute $\mathbf{L}$ with

$\mathbf{L}\mathbf{L}^{\mathrm{T}} = \mathbf{I}_n + \frac{\sigma}{2}\nabla^2 p(\mathbf{x}_{j+1}(\frac{\sigma}{2}))$.

Case 1. $\mathbf{I}_n + \frac{\sigma}{2}\nabla^2 p(\mathbf{x}_{j+1}(\frac{\sigma}{2}))$ is not positive definite

$\sigma := 0.5\sigma$

goto Step 3.

Case 2. $\mathbf{I}_n + \frac{\sigma}{2}\nabla^2 p(\mathbf{x}_{j+1}(\frac{\sigma}{2}))$ is positive definite
goto Step 9.

Step 9 (Computation of $\mathbf{x}^*_{j+1}(\sigma)$).
Compute $\mathbf{x}^*_{j+1}(\sigma)$ via
$\mathbf{L}\mathbf{L}^{\mathrm{T}}\mathbf{x}^*_{j+1}(\sigma) = \frac{\sigma}{2}\nabla p(\mathbf{x}_{j+1}(\frac{\sigma}{2})) - \varepsilon\sqrt{\sigma/2}\,\mathbf{n}_2$ and
$\mathbf{x}^*_{j+1}(\sigma) := \mathbf{x}_{j+1}(\frac{\sigma}{2}) - \mathbf{x}^*_{j+1}(\sigma)$,
goto Step 10.
$\bar{\mathbf{X}}^{\varepsilon}\left(\sigma; \bar{\mathbf{X}}^{\varepsilon}\left(\frac{\sigma}{2}; \mathbf{x}_j, \frac{\sigma}{2}, \omega^n\right), \frac{\sigma}{2}, \omega^n\right)$ *is denoted by* $\mathbf{x}^*_{j+1}(\sigma)$.

Step 10 (Acceptance condition).
Case 1. $\|\mathbf{x}_{j+1}(\sigma) - \mathbf{x}^*_{j+1}(\sigma)\|_2 \le \delta$
$\mathbf{x}_{j+1} := \mathbf{x}^*_{j+1}(\sigma)$,
print$(j+1, \mathbf{x}_{j+1}, p(\mathbf{x}_{j+1}))$,
goto Step 1.
Case 2. $\|\mathbf{x}_{j+1}(\sigma) - \mathbf{x}^*_{j+1}(\sigma)\|_2 > \delta$
$\sigma := \frac{\sigma}{2}$,
goto Step 3.

5. Implementation and numerical results. In this section we consider a parallel implementation on a transputer network of the algorithm formulated above. The transputer T805-25 from INMOS is a 32-bit microprocessor with a 64-bit floating point unit (FPU), graphics support, and 4Kbytes on-chip RAM. This processor is able to perform 1.875 Mflops at a processor speed of 25 MHz. A transputer has a configurable memory interface and four bidirectional communication links that allow networks of transputers constructed by direct point-to-point connections without external logic. Each link can transfer data at up to 2.35 Mbytes/s. By way of the external memory interface the processor has access to maximal 4Gbytes of memory. In a transputer the FPU, the CPU, and the data transfer work physically in parallel (for details see [12]). For the transputer architecture just described it is advantageous to interpret the software as a set of processes that can be performed in serial or in parallel. After starting execution, a process performs a number of actions and then either terminates or stops. Each action may be an assignment, an input, or an output. The communication between two processes is achieved by means of channels, where a channel between processes executing on the same transputer is implemented by a single word in the memory, and a channel between different transputers is implemented by point-to-point links. Transputers can be programmed in most high-level languages. Using the standard languages it is not possible to obtain all advantages of the transputer architecture (especially if concurrency is required). Hence, the high-level language OCCAM 2 was designed in tandem with the transputer. Every transputer implements the OCCAM 2 concepts of concurrency and communication. OCCAM 2 can be used to program an individual transputer or to program a network of transputers. When OCCAM 2 is used to program an individual transputer, the transputer shares its time between the concurrent processes, and channel communication is implemented by moving data within the memory. Communication between OCCAM 2 processes on different transputers is implemented directly by transputer links. Details about the language OCCAM 2 can be found in [11].

The algorithm described in §4 was implemented on a transputer network with 27 transputers. A suitable network topology was chosen to implement an efficient parallel version of the Cholesky factorization (cf. [3]) and an efficient parallel version of automatic differentiation (cf. [6]–[8]). Moreover, some steps of the algorithm, e.g., steps 3 and 5, and various paths of $\{\mathbf{X}_t^{\varepsilon,\mathbf{x}_0}\}$, can be computed in parallel.

TABLE 1
Numerical results.

Problem	1	2	3	4	5	6	7
Dimension n	80	90	100	110	120	100	100
δ	0.1	0.1	0.1	0.1	0.1	0.1	0.1
ε	2	2	2	2	2	3	3
$p(\mathbf{x}_{\text{lo}})$	4.4E10	7.1E10	1.3E11	1.2E11	2.2E11	5.4E7	5.0E7
$p(\mathbf{x}_{\text{s}})$	72.1	93.5	107.3	125.8	136.7	0.19	0.23
$F(\mathbf{x}_{\text{gl}})$	1.5E–5	7.3E–5	3.8E–5	5.6E–6	8.3E–5	6.1E–6	21.E–5
# it	1000	1000	1000	1000	1000	1000	1000
# ∇	8085	8182	7891	7697	8376	2112	2296
# ∇^2	8085	8182	7891	7697	8376	2112	2296
# Ch	15170	15364	14782	14394	15752	3224	3592

Since each path of $\{\mathbf{X}_t^{\varepsilon,\mathbf{x}_0}\}$ is a function $[0,\infty[\to \mathbb{R}^n$, our algorithm does not terminate. Each computed point $\mathbf{x}_j$ can be used as a starting point for a local minimization procedure that works parallel to the above algorithm. For the numerical results we determined a fixed number of iterations for each example. The computed point with the smallest function value of p was chosen as a starting point for the local minimization.

Now we consider the numerical results of seven test problems (see Table 1). The first five problems are complementarity problems that are very important in parametric nonlinear programming, for instance, compare [13] and [14]. We investigate special complementarity problems of the following type.

Compute $\mathbf{x}^* \in \mathbb{R}^n$ for given $\mathbf{q} \in \mathbb{R}^n$, $\mathbf{M} \in \mathbb{R}^{n,n}$ such that

$$\begin{aligned}(\mathbf{q}+\mathbf{M}\mathbf{x}^*)^{\mathrm{T}}\mathbf{x}^* &= 0,\\ x_i^* &\geq 0 \quad i=1,\dots,n,\\ (\mathbf{q}+\mathbf{M}\mathbf{x}^*)_i &\geq 0 \quad i=1,\dots,n.\end{aligned}$$

For the computation of $\mathbf{x}^*$, we solve a global optimization problem with objective function

$$p(\mathbf{x}) = \left(1+\left(\mathbf{q}^{\mathrm{T}}\mathbf{x}+\frac{1}{2}\mathbf{x}^{\mathrm{T}}(\mathbf{M}+\mathbf{M}^{\mathrm{T}})\mathbf{x}\right)^2\right)^{1/2} - 1 + 1000\sum_{i=1}^{n}\left(P(-x_i)^4 + P(-(\mathbf{q}+\mathbf{M}\mathbf{x})_i)^4\right).$$

The method of Best and Ritter [2] is chosen as the local minimization method for the quadratic problem (QP)

$$\text{(QP)} \qquad \min_{\mathbf{x}}\{F(\mathbf{x}) := \mathbf{q}^{\mathrm{T}}\mathbf{x}+\frac{1}{2}\mathbf{x}^{\mathrm{T}}(\mathbf{M}+\mathbf{M}^{\mathrm{T}})\mathbf{x} \mid x_i \geq 0 \quad i=1,\dots,n, \quad (\mathbf{q}+\mathbf{M}\mathbf{x})_i \geq 0 \quad i=1,\dots,n\}.$$

The special choice of the penalty function p in contrast to those in §4 is motivated by the following example:

$$\min_{\mathbf{x}}\{F(\mathbf{x}) := x_1 + x_1^2 - x_1x_2 \mid x_i \geq 0,\ i=1,2,\ x_1 - x_2 + 1 \geq 0\}.$$

Using $\mathbf{x}(s) = \binom{-2}{0} + s\binom{1}{1}$, $s \geq 2$, we obtain

$$F(\mathbf{x}(s)) = 2 - s,$$

$$\mu \sum_{i=1}^{2} (P(-x_i(s))^4 + P(-(\mathbf{q} + \mathbf{M}\mathbf{x}(s))_i)^4) \equiv \mu.$$

Hence, the penalty functions p defined in §4 are unbounded from below for all $\mu \geq 0$.

For $n = 80$, 90, 100, 110, and 120, the complementarity problems (test problems 1–5) are generated in the following way.

Choose $x_i = 5 \quad i = 20k + 1, \ldots, 20k + 10,\ k \in \mathbb{N}_0,\ k \leq 0.05(n - 10)$.

Choose $x_i = 0 \quad i = 20k + 11, \ldots, 20k + 20,\ k \in \mathbb{N}_0,\ k \leq 0.05(n - 20)$.

Compute $\mathbf{M} \in \mathbb{R}^{n,n}$ using a random number generator.

Compute $q_i = -(\mathbf{M}\mathbf{x})_i$ for $x_i > 0$ or ($x_i = 0$ and i odd).

Compute $q_i > -(\mathbf{M}\mathbf{x})_i$ for $x_i = 0$ and i even using positive random numbers.

The first and second derivatives of p are computed analytically. Test problems 6 and 7 are global minimization problems with equality constraints.

Problem 6.

$$\min_{\mathbf{x}} \Bigg\{ F(\mathbf{x}) := 1 + 6\left(x_1 - \sum_{i=2}^{100} x_i\right)^2 - \cos\left(12\left(x_1 - \sum_{i=2}^{100} x_i\right)\right) + 1000 \sum_{i=2}^{100} x_i^2 \;\Bigg|\; \sum_{i=6}^{100} x_i^2 - x_5^2 = 0, \quad x_2^2 + x_4^2 - x_3^2 = 0 \Bigg\}.$$

This problem has five local minimizers. The unique global minimizer is at $\mathbf{x}^* = \mathbf{0}$, $F(\mathbf{x}^*) = 0$.

Problem 7.

$$\min_{\mathbf{x}} \Bigg\{ F(\mathbf{x}) := 2 + 6\left(x_1 - \sin\left(\sum_{i=3}^{100} x_i\right)\right)^2 - \cos\left(12\left(x_1 - \sin\left(\sum_{i=3}^{100} x_i\right)\right)\right) + 1000 \sum_{i=3}^{100} x_i^2 + 6\left(x_2 - \sin\left(\sum_{i=3}^{100} x_i\right)\right)^2 - \cos\left(12\left(x_2 - \sin\left(\sum_{i=3}^{100} x_i\right)\right)\right) \;\Bigg|\; \sum_{i=8}^{98} x_i^2 - x_6^2 = 0, \quad x_{99}^2 - x_{100}^2 = 0, \quad x_7^2 + x_3^2 + x_5^2 - x_4^2 = 0 \Bigg\}.$$

This problem has 25 local minimizers. The unique global minimizer is at $\mathbf{x}^* = \mathbf{0}$, $F(\mathbf{x}^*) = 0$.

The type of penalty functions p introduced in §4 is used for the numerical calculations, namely,

$$p(\mathbf{x}) = F(\mathbf{x}) + 1000 \sum_{i=1}^{q} h_i^2(\mathbf{x}).$$

The local minimization is done by an SQP method; see [17]. The derivatives of these problems are computed using automatic differentiation. Each Cholesky factorization of a positive definite matrix with dimension 120 (highest dimension of the test problems) needs 0.3 sec

computing time using the above-mentioned parallel implementation. The computing time for automatic differentiation depends on the complexity of the objective function. As a starting point of our algorithm, a point $\mathbf{x}_{\text{lo}}$ far away from a global minimizer of the given problem is chosen. After a fixed number of iterations, the computed point with the smallest function value of p (denoted by $\mathbf{x}_{\text{s}}$) is chosen as the starting point for the local minimization. The result of this local minimization is denoted by $\mathbf{x}_{\text{gl}}$. The following quantities concerning the above-mentioned algorithm are listed (without the local minimizations).

δ, ε,

$p(\mathbf{x}_{\text{lo}})$, $p(\mathbf{x}_{\text{s}})$, $F(\mathbf{x}_{\text{gl}})$,

it [chosen number of iterations (equal to the number of function evaluations)],

∇ [number of gradient evaluations],

∇^2 [number of Hessian matrix evaluations],

Ch [number of Cholesky decompositions].

The solutions of (CGOP) were obtained for all seven problems after a few steps with a local minimization procedure (cf. [2] for problems 1–5 and [17] for problems 6 and 7) using $\mathbf{x}_{\text{s}}$ as the starting point. Further numerical investigations show that the number of iterations that are necessary to compute a point close enough to a global minimizer increase rapidly with the number of local minimizers of the penalty function, but not with the dimension of the problem.

REFERENCES

[1] H. Bauer, *Wahrscheinlichkeitstheorie*, de Gruyter, Berlin, New York 1991.

[2] M. J. Best and K. Ritter, *A quadratic programming algorithm*, ZOR, 32 (1988), pp. 271–297.

[3] F. Dandl, M. Hogger, and S. Schäffler, *A Parallel Implementation of the Cholesky Factorization of Positive Definite Matrices on a Transputer Network*, Tech. Report No. IAMS1993.1TUM, Technical University of Munich, Germany, 1993.

[4] L. C. W. Dixon and G. P. Szegö, *Towards Global Optimization*, North-Holland, Amsterdam, 1975.

[5] ———, *Towards Global Optimization* 2, North-Holland, Amsterdam, 1978.

[6] H. Fischer, *Some aspects of automatic differentiation*, in Numerical Methods and Approximation Theory III, Proc. 3rd Internat. Conf. on Numerical Methods and Approximation Theory, G. V. Milovanovic, ed., University of Nis, Yugoslavia, August 18–21, 1987, 1988, pp. 199–208.

[7] H. Fischer, *Automatic differentiation of characterizing sequences*, J. Comput. Appl. Math., 28 (1989), pp. 181–185.

[8] ———, *Automatic differentiation: Parallel computation of function, gradient, and Hessian matrix*, Parallel Comput., 13 (1990), pp. 101–110.

[9] S. B. Gelfand and S. K. Mitter, *Metropolis-type annealing algorithms for global optimization in* $\mathbb{R}^n$, SIAM J. Control Optim., 31 (1993), pp. 111–131.

[10] R. Z. Hasminskij, *Stochastic Stability of Differential Equations*, Sijthoff and Noordhoff, Amsterdam, 1980.

[11] Inmos, *OCCAM 2 Reference Manual*, Prentice-Hall, Englewood Cliffs, NJ, 1988.

[12] Inmos, *Transputer Reference Manual*, Prentice-Hall, Englewood Cliffs, NJ, 1988.

[13] M. Kojima, *A complementary pivoting approach to parametric nonlinear programming*, Math. Oper. Res., 4 (1979), pp. 464–477.

[14] N. Megiddo and M. Kojima, *On the existence and uniqueness of solutions in nonlinear complementarity theory*, Math. Programming, 12 (1977), pp. 110–130.

[15] P. Protter, *Stochastic Integration and Differential Equations*, Springer-Verlag, Berlin, New York, 1990.

[16] S. Schäffler, *Global optimization using stochastic integral equations*, J. Global Optim., submitted.

[17] K. Schittkowski, *The nonlinear programming method of Wilson, Han and Powell with an augmented Lagrangian type line search function*, Numer. Math., 38 (1981), pp. 83–114.

[18] A. Törn and A. Žilinskas, *Global Optimization*, Springer-Verlag, Berlin, Heidelberg, New York, 1987.