TECHNICAL RESEARCH REPORT

SYSTEMS RESEARCH CENTER

Supported by the National Science Foundation Engineering Research Center Program (NSFD CD 8803012), Industry and the University

On Combining Feasibility, Descent and Superlinear Convergence in Inequality Constrained Optimization

by E.R. Panier and A.L. Tits

TR 89-27r2

To appear in Mathematical Programming

On Combining Feasibility, Descent and Superlinear Convergence in Inequality Constrained Optimization [1]

Eliane R. Panier[2] *and André L. Tits*

Systems Research Center

and

Electrical Engineering Department

University of Maryland, College Park, MD 20742

Abbreviated title: Feasibility, Descent and Superlinear Convergence

Abstract. Extension of quasi-Newton techniques from unconstrained to constrained optimization via Sequential Quadratic Programming (SQP) presents several difficulties. Among these are the possible inconsistency, away from the solution, of first order approximations to the constraints, resulting in infeasibility of the quadratic programs; and the task of selecting a suitable merit function, to induce global convergence. In the case of inequality constrained optimization, both of these difficulties disappear if the algorithm is forced to generate iterates that all satisfy the constraints, and that yield monotonically decreasing objective function values. (Feasibility of the successive iterates is in fact required in many contexts such as in real-time applications or when the objective function is not well defined outside the feasible set). It has been recently shown that this can be achieved while preserving local two-step superlinear convergence. In this note, the essential ingredients for an SQP-based method exhibiting the desired properties are highlighted. Correspondingly, a class of such algorithms is described and analyzed. Tests performed with an efficient implementation are discussed.

Key words: constrained optimization, sequential quadratic programming, feasibility, superlinear convergence.

AMS(MOS) subject classifications: 90C30, 65K10.

[1] This research was supported in part by NSF's Engineering Research Centers Program No. NSFD-CDR-88-03012, and by NSF grants No. DMC-84-51515 and DMC-88-15996.

[2] Present address: Chase Manhattan Bank, 900 Stewart Avenue, Garden City, New York 11530.

1. Introduction

Consider the optimization problem

$$(P) \quad \min \ f(x) \quad \text{s.t. } x \in X$$

where $X = \{x \text{ s.t. } g(x) \leq 0\}$, with $f : \mathbb{R}^n \to \mathbb{R}$ and $g : \mathbb{R}^n \to \mathbb{R}^m$ smooth functions.

In its simplest version, given estimates x of the solution of (P) and H of the Hessian of the corresponding Lagrangian function, the Sequential Quadratic Programming (SQP) iteration consists of first computing a search direction d^0 by solving the quadratic program

$$\begin{aligned} \min_{d^0} \quad & \frac{1}{2}\langle d^0, Hd^0\rangle + \langle \nabla f(x), d^0\rangle \\ \text{s.t.} \quad & g_j(x) + \langle \nabla g_j(x), d^0\rangle \leq 0, \quad j = 1, \ldots m \end{aligned} \tag{1.1}$$

and then performing a search along d^0. Two difficulties (among others) that arise in connection with this iteration are (i) the possible inconsistency of the constraints in (1.1) and (ii) the problem of selecting a suitable criterion for the line search (essentially, a suitable merit function). These questions have been the object of intense research and various avenues have been proposed to address them (see, e.g., [2], [3], [5], [7], [12], [14]).

In this note we explore the point of view that, if the algorithm is somehow forced to construct iterates x that satisfy the constraints, then (1.1) will always be feasible and it will be possible to base the line search on a decrease of the objective function f itself, thus addressing both of the aforementioned difficulties. Accordingly, we wish to enforce on the sequence of iterates $\{x_k\}$ the conditions

$$x_k \in X \tag{1.2}$$

$$f(x_{k+1}) \leq f(x_k)\ . \tag{1.3}$$

These two conditions (especially (1.2)) turn out to be important in their own right in many contexts, e.g., (i) when the objective function is not well defined outside the feasible set or (ii) in real-time applications, when it is crucial that a feasible solution be available at the next "stopping time". It was recently shown that it is indeed possible to satisfy (1.2)

Section 2. Convergence results are briefly reported in Section 3. Implementation details as well as some numerical results are discussed in Section 4. Section 5 is devoted to concluding remarks. Many of the results given in Section 3 are stated without proof as they are essentially identical to results in [10]. To avoid any loss of continuity, other proofs are given in an appendix.

2. A class of algorithms

The following is assumed to hold.

A1. The feasible set X is nonempty.

A2. The functions $f,\ g_j,\ j = 1, \ldots, m$ are continuously differentiable.

A3. For all $x \in X$, the vectors $\{\nabla g_j(x),\ j \in I(x)\}$ are linearly independent, where

$$I(x) = \{j \mid g_j(x) = 0\}.$$

A point x^* is said to be a *Karush-Kuhn-Tucker (KKT)* point for (P) if $x^* \in X$ and there exit some nonnegative multipliers $\mu_j^*,\ j = 1, \ldots m$ satisfying

$$\nabla_x L(x^*, \mu^*) = 0$$

and

$$\mu_j^* g_j(x^*) = 0, \quad j = 1, \ldots m$$

where $L(x, \mu)$ denotes the Lagrangian function

$$L(x, \mu) = f(x) + \sum_{j=1}^{m} \mu_j g_j(x).$$

The point x^* is said to satisfy *second order sufficiency conditions with strict complementary slackness* if (i) the multipliers satisfy $\mu_j^* > 0,\ \forall\ j \in I(x^*)$ and (ii) $f, g_j, i = 1, \ldots,\ m$ are twice continuously differentiable and the Hessian of the Lagrangian function $\nabla_{xx} L(x^*, \mu^*)$ is positive definite on the subspace $\{p \mid \langle \nabla g_j(x^*), p \rangle = 0,\ \forall\ j \in I(x^*)\}$.

As indicated in the introduction three quantities used in Algorithm 2.1 below will be unspecified but subject to certain restrictions. First, the feasible descent direction d^1 will be obtained for the current iterate x via a continuous map $d^1(\cdot) : \mathbb{R}^n \to \mathbb{R}^n$ satisfying

$$d^1(x) = 0 \text{ if } x \text{ is a KKT point} \tag{2.1}$$

$$\langle \nabla f(x), d^1(x) \rangle < 0 \text{ if } x \text{ is not a KKT point} \tag{2.2}$$

and

$$\langle \nabla g_j(x), d^1(x) \rangle < 0 \text{ if } x \text{ is not a KKT point and } j \in I(x). \tag{2.3}$$

(Continuity is only assumed for sake of simplicity; the results hold with milder assumptions.) Such a direction can, for example, be obtained as the solution of

$$\min_{d^1} \quad \frac{1}{2}\|d^1\|^2 + \max\{\langle \nabla f(x), d^1 \rangle;\ \max_j\{g_j(x) + \langle \nabla g_j(x), d^1 \rangle\}\ \}. \tag{2.4}$$

Second the coefficient ρ in the convex combination determining d will be computed, given d^0, via a map $\rho(\cdot) : \mathbb{R}^n \to [0,1]$ such that $\rho(d^0) = 1$ (so that $d = d^1$) when $\|d^0\|$ is larger than some given threshold, $\rho(d^0)$ is bounded away from zero outside every neighborhood of zero and, for $\|d^0\|$ small,

$$\rho(d^0) = O(\|d^0\|^2)\ . \tag{2.5}$$

For example, for $\nu \geq 2$, $\min(1, \|d^0\|^\nu)$ and $\|d^0\|^\nu/(1 + \|d^0\|^\nu)$ (with the value 1 if $\|d^0\|$ is very large) both satisfy these conditions. While we had merely indicated that ρ should go to 0 as $\|d^0\|$ does, the more stringent condition (2.5) will be necessary for the line search along $x + td + t^2\tilde{d}$ to asymptotically yield a step of one. Finally, the goal of correction $\tilde{d}$ is to make $x + d + \tilde{d}$ a suitable next iterate when x is close to a solution. Thus it must be small compared to d while ensuring $g(x + d + \tilde{d}) < 0$ and $f(x + d + \tilde{d}) < f(x)$. Given a current iterate x, a direction d and a positive definite estimate H of the Hessian of the Lagrangian, we select $\tilde{d} = \tilde{d}(x, d, H)$, defined to be the solution of the quadratic program

$$\begin{aligned} \min_{\tilde{d}} \quad & \frac{1}{2}\langle d + \tilde{d}, H(d + \tilde{d}) \rangle + \langle \nabla f(x), d + \tilde{d} \rangle \\ \text{s.t.} \quad & g_j(x + d) + \langle \nabla g_j(x), \tilde{d} \rangle \leq -\|d\|^\tau, \quad j = 1, \ldots, m \end{aligned} \tag{2.6}$$

if it exists and has norm less than $\min\{\|d\|, C\}$, where C is a given large number, and $\tilde{d}(x, d, H) = 0$ otherwise. Here, $\tau \in$(2,3) is preselected. Note that (2.6) will likely yield better results than, say,

$$\begin{aligned} \min_{\tilde{d}} \quad & \frac{1}{2}\|\tilde{d}\|^2 \\ \text{s.t.} \quad & g_j(x + d) + \langle \nabla g_j(x), \tilde{d} \rangle = -\|d\|^\tau, \quad \forall j \in \{j \mid g_j(x) + \langle \nabla g_j(x), d^0 \rangle = 0\}, \end{aligned}$$

inspired by [9] and used in [10], since the former uses a (refined) model of (P) rather than merely a refined model of the active constraints.

Algorithm 2.1

Parameters. $\alpha \in (0, \frac{1}{2})$, $\beta \in (0,1)$.

Data. $x_0 \in X$, $H_0 \in \mathbb{R}^{n\times n}$, symmetric positive definite.

Step 0. Initialization. Set $k = 0$.

Step 1. Computation of a search arc.

i. Compute d_k^0 by solving the quadratic program

$$\begin{aligned} \min_{d^0} \quad & \frac{1}{2}\langle d^0, H_k d^0\rangle + \langle \nabla f(x_k), d^0\rangle \\ \text{s.t.} \quad & g_j(x_k) + \langle \nabla g_j(x_k), d^0\rangle \le 0, \quad j = 1,\dots m. \end{aligned} \tag{2.7}$$

If $d_k^0 = 0$ stop.

ii. Let $d_k^1 = d^1(x_k)$, $\rho_k = \rho(d_k^0)$ and set $d_k = (1-\rho_k)d_k^0 + \rho_k d_k^1$.

iii. Let $\tilde{d}_k = \tilde{d}(x_k, d_k, H_k)$.

Step 2. Arc search. Compute t_k, the first number t in the sequence $\{1, \beta, \beta^2, \cdots\}$ satisfying

$$f(x_k + td_k + t^2\tilde{d}_k) \le f(x_k) + \alpha t\langle \nabla f(x_k), d_k\rangle \tag{2.8}$$

$$g_j(x_k + td_k + t^2\tilde{d}_k) \le 0, \quad j = 1,\dots m. \tag{2.9}$$

Step 3. Updates. Compute a new symmetric positive definite approximation H_{k+1} to the Hessian of the Lagrangian. Set $x_{k+1} = x_k + t_k d_k + t_k^2 \tilde{d}_k$. Set $k = k+1$. Go back to Step 1. □

3. Convergence results

3.1. Global convergence

In order to prove convergence of Algorithm 2.1, we will assume that the matrices H_k are bounded, i.e.,

$$\|H_k\| \le \bar{H}, \quad \forall\, k \tag{3.1}$$

for some $\tilde{H} > 0$. The first result is that the algorithm is well defined. This follows from Propositions 3.1 and 3.2 below.

Proposition 3.1. Direction d_k^0 is always well defined and is equal to zero if, and only if, x_k is a KKT point for (P). In particular, if Algorithm 2.1 stops at Step 1.*i*, then x_k is a KKT point. If x_k is not a KKT point, d_k^0 satisfies

$$\langle \nabla f(x_k), d_k^0 \rangle < 0 \tag{3.2}$$

and

$$\langle \nabla g_j(x_k), d_k^0 \rangle \leq 0, \quad \forall\, j \in I(x_k). \tag{3.3}$$

□

Assume now that the algorithm never stops in Step 1.*i*. The next result is standard for feasible descent directions. In view of (3.2)-(3.3) and of the properties of $d^1(\cdot)$, it applies here since $\rho(d^0) > 0$ whenever $d^0 \neq 0$.

Proposition 3.2. The line search yields a step $t_k = \beta^j$ for some finite $j = j(k)$. □

Establishing global convergence presents no conceptual difficulty.

Proposition 3.3. Let x^* be an accumulation point of the sequence generated by Algorithm 2.1. Then, x^* is a KKT point for (P). □

3.2. Rate of convergence

We now assume some additional regularity for the functions involved. Assumption A2 is replaced by the following.

A2′. The functions f and g_j, $j = 1, \ldots m$ are three times continuously differentiable.

Assumptions A1 and A3 are still assumed to hold. We also assume that there exists a scalar $\underline{H} > 0$ such that, for all k, the Hessian estimates satisfy

$$\langle d, H_k d \rangle \geq \underline{H} \|d\|^2, \quad \forall\, d \in \mathbb{R}^n. \tag{3.4}$$

Under these assumptions, the following can be shown.

Proposition 3.4. If some accumulation point x^* of the sequence generated by Algorithm 2.1 satisfies the second order sufficiency conditions with strict complementary slackness, then the entire sequence converges to x^*. □

In the sequel, we assume that the sequence generated by the algorithm converges to such a point x^*. As a first consequence of this, the search direction converges to zero, the multipliers converge to μ^* and the active constraints are eventually correctly identified. Specifically, the following holds.

Proposition 3.5.

i) *a)* $\{d_k^0\} \to 0$, *b)* $\{d_k^1\} \to 0$, *c)* $\{d_k\} \to 0$.

ii) $\{\mu_k\} \to \mu^*$.

iii) For k large enough,

$$\{j \mid g_j(x_k) + \langle \nabla g_j(x_k), d_k^0 \rangle = 0\} = \{j \mid \mu_{k,j} > 0\} = I(x^*).$$

where μ_k is the multiplier vector associated with the constraints in (2.7). □

A crucial requirement, in order to obtain superlinear convergence, is that a unit step size be used in a neighborhood of the solution. This is achieved here thanks to the correction $\tilde{d}_k$, provided H_k suitably approximates the Hessian of the Lagrangian at x^*. Specifically, let us assume that

$$\frac{\|P_k(H_k - \nabla^2_{xx} L(x^*, \mu^*))P_k d_k\|}{\|d_k\|} \to 0 \quad \text{as } k \to \infty \tag{3.5}$$

where μ^* is the multiplier vector associated with x^* and where the matrices P_k are defined by

$$P_k = I - R_k(R_k^T R_k)^{-1} R_k^T$$

with $R_k = [\nabla g_j(x_k) \mid j \in I(x^*)] \in \mathbb{R}^{n \times |I(x^*)|}$. (Note that, in view of the linear independence of the gradients of the active constraints at x^*, the matrices $R_k^T R_k$ are invertible for k large enough.) This holds, for example, under some conditions, when H_k is constructed using the BFGS update formula (see [13]).

Proposition 3.6. For k large enough, the step size t_k is one. □

Finally, two-step superlinear convergence follows. The proof is not given as it follows step by step, with minor modifications, that of [13, Sections 2-3] (see also [16]).

Theorem 3.7. Under the stated assumptions, the convergence is two-step superlinear, i.e.,

$$\lim_{k\to\infty} \frac{\|x_{k+2} - x^*\|}{\|x_k - x^*\|} = 0.$$

□

4. Implementation and Computational Experiments

An efficient implementation of Algorithm 2.1 has been produced as part of a FORTRAN code dubbed FSQP (available from the authors). Version 2.3 [17] of FSQP was used to perform the numerical tests discussed below. In FSQP, d_k^1 is the minimizer for the quadratic program (all norms are Euclidean norms)

$$\min_{d^1\in R^n} \frac{0.1}{2}\langle d_k^0 - d^1, d_k^0 - d^1\rangle + \max\{\langle\nabla f(x_k), d^1\rangle; \max_j\{g_j(x_k) + \langle\nabla g_j(x_k), d^1\rangle\}\}, \quad (5.1)$$

ρ_k is given by

$$\rho_k = \frac{\|d_k^0\|^{2.1}}{\|d_k^0\|^{2.1} + \max(0.5, \|d_k^1\|^{2.5})}, \quad (5.2)$$

and $\tilde{d}_k$ is the minimizer for the quadratic program

$$\begin{aligned} &\min_{\tilde{d}\in R^n} \frac{1}{2}\langle(d_k + \tilde{d}), H_k(d_k + \tilde{d})\rangle + \langle\nabla f(x_k), \tilde{d}\rangle \\ &\text{s.t. } g_j(x_k + d_k) + \langle\nabla g_j(x_k), \tilde{d}\rangle \le -\min(0.01\|d_k\|, \|d_k\|^{2.5}), \ j \in I_k \end{aligned} \quad (5.3)$$

where I_k includes the indexes of all constraints for which the associated multiplier in (2.7) is positive, as well as those of all constraints for which

$$g_j(x_k) \ge -0.1\|\nabla g_j(x_k)\| \cdot \|d_k^0\|.$$

An advantage of (5.1) over (2.4) is that, if the former is used, d_k^1 tends to stay close to the quasi-Newton direction d_k^0 (and dependence of d_k^1 on d_k^0 does not affect the analysis). Formula (5.2) for ρ_k is preferred to the formulae given after (2.5) because d_k is then less sensitive to large values of the norm of d_k^1 (and again, dependence on d_k^1 does not affect the analysis). Finally, we have used (5.3) instead of (2.6) because the former prevents an excessive correction when d_k is large, and constraints unlikely to affect $\tilde{d}_k$ are not evaluated. FSQP updates H_k according to the BFGS formula with Powell's modification [12] and H_0 is the identity matrix; and α and β have values 10^{-7} and 0.5 respectively. Next, in our tests, the quadratic programming subproblems (2.7), (5.1) and (5.3) were solved by means of subroutine QPSOL Version 3.2 [6], with the "feasibility tolerance" parameter set to 10^{-14}. Concerning the line search, FSQP 2.3 pays special attention to the order in which the functions (objective and constraints) are evaluated. For $t = 1$, the constraints are evaluated before the objective and in the order of their indexes, except that all constraints for which the KKT multiplier in (2.7) is nonzero are evaluated first (and the tests are terminated as soon as a violation is detected); for $t < 1$, the same rule is used, except that the function (objective or constraint) whose violation led to test termination at the previous trial step is evaluated first. (Note that such a strategy would not be possible with a penalty function-based line search.) Finally, FSQP 2.3 includes provisions to efficiently handle affine constraints, and affine equality constraints are also allowed. Specifically in (2.7), (5.1) and (5.3), affine feasibility is required for $x_k + d^0$, $x_k + d^1$, and $x_k + d_k + \tilde{d}$, respectively.

Experiments were conducted on all problems from [8] where a feasible initial point is provided and no nonlinear equality constraints are present. We tested three algorithms on those problems: Algorithm 2.1 as implemented in FSQP 2.3, Algorithm A of [10], and the 1984 version of VF02AD [1]. In all cases the stopping criterion was that the Euclidean norm of the gradient of the Lagrangian (KKT vector), with KKT multipliers corresponding to quadratic program (2.7), be less than a prespecified small ϵ (the value of this threshold was chosen, for each problem, based on the norm of the final KKT vector as reported in [8]). The results are summarized in Table 1. (Some results for Algorithm A are slightly different from those reported in [10] because of a different stopping criterion.) In that table, No is the number of the test problem in [8], NF is the number of evaluations of the objective function, NG the number of evaluations of scalar constraint functions, NIT the number of iterations, FV the final value of the objective function, VC the final constraint violation (always 0 for FSQP and A), KKT the Euclidean norm of the final KKT vector, and EPS the stopping criterion threshold (ϵ). Finally an asterisk ($*$) indicates that Algorithm A was unable to proceed except possibly by performing a first order iteration, and a pound sign (#) indicates termination due to failure of the line search (t smaller than the machine epsilon). All tests were performed in double precision, on a Sun Microsystems Sparcstation 1.

Locally the iterations for all three algorithms are essentially identical. The main difference between Algorithm 2.1 and Algorithm A of [10] is a simpler, more rational scheme for selecting the search direction away from a solution. On this basis, it was hoped that FSQP would improve on Algorithm A of [7] in many of the instances when the number of objective function evaluations of the latter seemed unduly large when compared to that of VF02AD, e.g., for problems 57, 100 and 117. It appears that this is indeed the case. Also note that FSQP typically outperforms Algorithm A in term of number of constraint evaluations. Comparison of the results achieved by VF02AD and FSQP gives an indication of the "cost of feasibility". While, in terms of number of objective function evaluations, VF02AD and FSQP appear to be roughly comparable, the number of constraint function evaluations, as could be expected, is often larger for FSQP than for VF02AD. This is especially so when most constraints are active at the solution, as is the case for problems 12, 29, 30, 31, 34, 66, 93 and 117 (but note that FSQP requires fewer constraints evaluations than VF02AD in problems 67, 84 and 85, for which only one or two constraints are active at the solution). It is clearly due to the auxiliary constraint evaluations at points $x_k + d_k$. This certainly is an aspect potential users should take into account. Independently of the number of constraint evaluations, the work per iteration in FSQP is larger than that in VF02AD, as three quadratic programs are solved instead of one. This difference may not be of concern however in situations where the computer time necessary to solve a quadratic program is small compared to that needed to perform a function evaluation, as is often the case in engineering applications. Finally, a peculiar phenomenon occurs with problem 67. The final objective value achieved by VF02AD is larger than the initial value ($-0.868725652E+03$)! Such undesirable phenomenon cannot occur with a feasible, descent method.

5. Concluding remarks

We have shown how quasi-Newton SQP-type algorithms generating feasible iterates with monotone decrease of the objective function can be constructed based on some simple rules. The search direction d was obtained by perturbing the SQP direction d^0 with a feasible descent direction d^1. It should be noted that, alternatively, one could use for d^1 a mere feasible direction, not necessarily of descent for f. In constructing the search direction $d = (1 - \rho)d^0 + \rho d^1$, one should then select ρ small enough for the additional condition

$$\langle \nabla f(x), d \rangle \leq \theta \langle \nabla f(x), d^0 \rangle$$

to be satisfied, with $\theta \in (0, 1)$ a fixed number.

Another quasi-Newton method producing feasible iterates but involving only the solution of linear systems of equations (rather than quadratic programs) was recently proposed [11]. In the neighborhood of a solution the work per iteration of the latter and the SQP-type methods described here is similar, as the set of active constraints of the quadratic program eventually remains invariant from one iteration to the next. Away from the solution the tradeoff is that of a more accurate model of the problem (a quadratic program), likely to yield more efficient steps, versus a reduced amount of work per iteration.

Besides the FSQP Fortran batch implementation, Algorithm 2.1 has been integrated in an interactive optimization-based design package (C code: CONSOLE [4]). To this end, we had to devise an extension of the algorithm to constrained minimax and semi-infinite problems. Tests were performed on a variety of engineering system design problems, ranging from the design of a controller for a rotorcraft to the determination of appropriate temperature and feed flow profiles for a semi-batch copolymerization reactor. Remarkable success was achieved, reinforcing our belief that numerical optimization has a bright future in many areas of engineering design. In particular, we now feel that researchers in control systems engineering, who have not made significant use of numerical optimization in the past, will soon accept it as a viable tool.

Appendix. Proof of some propositions

We make use of the optimality conditions associated with the solution d_k of the quadratic program (2.7), namely

$$H_k d_k^0 + \nabla f(x_k) + \sum_{j=1}^{m} \mu_{k,j} \nabla g_j(x_k) = 0, \tag{A.1}$$

$$g_j(x_k) + \langle \nabla g_j(x_k), d_k^0 \rangle \leq 0, \quad j = 1, \ldots m, \tag{A.2}$$

$$\mu_{k,j}(g_j(x_k) + \langle \nabla g_j(x_k), d_k^0 \rangle) = 0, \quad j = 1, \ldots m, \tag{A.3}$$

for some nonnegative multipliers $\mu_{k,j}$, $j = 1, \ldots m$.

Proof of Proposition 3.1.

The claims follow easily from (A.1)-(A.3), positive definiteness of H_k and feasibility of x_k.

Proof of Proposition 3.3.

Let $\{x_k\}_{k \in K}$ be any subsequence converging to x^*. Two cases are to be considered.

i) The subsequence $\{d_k^0\}_{k \in K}$ converges to zero. In this case the result follows from an argument along the lines of the proof of Theorem 3.3 in [10].

ii) There exists a subsequence $\{d_k^0\}_{k \in K' \subset K}$ and a value $\underline{d}^0 > 0$ such that $\|d_k^0\| > \underline{d}^0$, $\forall\, k \in K'$. Suppose, by contradiction, that x^* is not a KKT point. Then, from the definition of the coefficients ρ_k and the fact that $\rho(\cdot)$ is bounded away from zero outside any neighborhood of zero, there exists a number $\underline{\rho} > 0$ such that $\rho_k \geq \underline{\rho}$, $\forall\, k \in K'$. Therefore, using (2.2) and (3.2) and the definition of d_k in Step 1.ii of Algorithm 2.1, we have

$$\langle \nabla f(x_k), d_k \rangle \leq \underline{\rho} \langle \nabla f(x_k), d_k^1 \rangle. \tag{A.4}$$

Also, in view of (2.7) and the feasibility of the iterates,

$$\langle \nabla g_j(x_k), d_k \rangle \leq -g_j(x_k) + \rho_k \langle \nabla g_j(x_k), d_k^1 \rangle, \quad j = 1, \ldots m\ .$$

Since, in view of (2.3), of Assumption A2 and of the continuity of $d^1(\cdot)$,

$$\langle \nabla g_j(x_k), d_k^1 \rangle < 0 \quad \forall j \in I(x^*)$$

for $k \in K'$ large enough, it follows that

$$\langle \nabla g_j(x_k), d_k \rangle \leq -g_j(x_k) + \underline{\rho} \langle \nabla g_j(x_k), d_k^1 \rangle, \quad \forall j \in I(x^*) \tag{A.5}$$

for $k \in K'$ large enough. In view of the contradiction assumption, of (A.4)-(A.5), and (2.2)-(2.3) and the continuity of $d^1(\cdot)$, there exists therefore a positive value $\underline{\delta}$ such that, for all $k \in K'$ large enough,

$$\langle \nabla f(x_k), d_k \rangle \leq -\underline{\delta},$$

$$\langle \nabla g_j(x_k), d_k \rangle \leq -\underline{\delta}, \text{ if } j \in I(x^*)$$

and, by continuity of g,

$$g_j(x_k) \leq -\underline{\delta}, \text{ if } j \notin I(x^*)$$

Also, since $d^1(\cdot)$ and $\tilde{d}(\cdot,\cdot,\cdot)$ are bounded on bounded sets and since $\rho(d^0) = 1$ for $\|d^0\|$ large enough, it follows from Step 1 *ii-iii* in Algorithm 2.1 that d_k and $\tilde{d}_k$ are bounded. The argument used in Proposition 3.2 of [10] then implies that, in this case, the step performed by the line search is bounded away from zero. This and the monotonic decrease of f imply therefore that the objective function is unbounded on $\{x_k\}_{k \in K'}$, in contradiction with the continuity of f. □

Proof of Proposition 3.4.

Under the stated assumptions, the KKT point x^* is isolated [15], i.e., for some $\epsilon > 0$, the ball $B(x^*, \epsilon)$ does not contain any KKT point other than x^*. Let $\{x_k\}_{k \in K}$ be any subsequence converging to x^*. Clearly, it is enough to show that

$$\|x_{k+1} - x_k\| \to 0, \; k \to \infty, k \in K. \tag{A.6}$$

Since $f(x_k)$ is monotonically decreasing, existence of an accumulation point of $\{x_k\}$ and continuity of f imply that the sequence $\{f(x_k)\}$ is bounded. Also, in view of the line search stopping criterion, we have

$$f(x_{k+1}) \leq f(x_k) + \alpha t_k \langle \nabla f(x_k), d_k \rangle.$$

Therefore, $t_k \langle \nabla f(x_k), d_k \rangle$ must tend to zero. Since x^* is a KKT point, continuity of $d^1(\cdot)$, boundedness of t_k and ρ_k and (2.1) imply that

$$t_k \rho_k d_k^1 \to 0, \; k \in K. \tag{A.7}$$

Also, in view of (A.1)-(A.2) and (3.4),

$$\langle \nabla f(x_k), d_k^0 \rangle \leq -\underline{H} \|d_k^0\|^2. \tag{A.8}$$

Since, in view of the definition of d_k in Step 1.ii of Algorithm 2.1,

$$\langle \nabla f(x_k), d_k \rangle = (1 - \rho_k)\langle \nabla f(x_k), d_k^0 \rangle + \rho_k \langle \nabla f(x_k), d_k^1 \rangle,$$

it follows that

$$t_k(1 - \rho_k)\underline{H}\|d_k^0\|^2 \rightarrow 0, \quad k \rightarrow \infty.$$

Since t_k and ρ_k are bounded, this implies that

$$t_k(1 - \rho_k)d_k^0 \rightarrow 0, \; k \in K.$$

In view of (A.7) and of the definition of d_k we conclude that $t_k d_k \rightarrow 0, k \in K$. Finally, since

$$\|x_{k+1} - x_k\| \leq t_k\|d_k\| + t_k^2\|\tilde{d}_k\| \leq 2t_k\|d_k\|,$$

the claim holds. □

Proof of Proposition 3.5.

Concerning claim *i) a)*, proceed by contradiction, i.e., suppose that $\|d_k^0\|$ is bounded away from 0 on a subsequence. Then, in view of (3.1) and (3.4), there exist some $K \subset \mathbb{N}$ and some positive definite matrix H^* such that $H_k \rightarrow H^*$ as $k \rightarrow \infty$, $k \in K$, and $\|d_k^0\|$ is bounded away from zero for $k \in K$. Using the same argument as in the proof of Proposition 4.2 in [10], it can then be proven that $d_k^0 \rightarrow 0$ as $k \rightarrow \infty$, $k \in K$, a contradiction. Thus *i) a)* holds. Claim *i) b)* holds due to property (2.1) and continuity of the search direction function $d^1(\cdot)$ and the fact that x^* is a KKT point. Finally, *i) c)* is satisfied in view of the definition of d_k, and of the properties of $\rho(\cdot)$. Parts *ii)* and *iii)* can be shown in the same way as in the proof of Proposition 4.2 in [10]. □

The following two lemmas are used below, in the proof of Proposition 3.6.

Lemma A.1. ([10, Lemma 4.4]) There exists some constant $\underline{C} > 0$ such that, for k large enough,

$$\sum_{j \in I(x^*)} \mu_{k,j} g_j(x_k) \;\leq\; -\underline{C}\Big(\sum_{j \in I(x^*)} g_j(x_k)^2\Big)^{\frac{1}{2}}.$$

□

Lemma A.2. The search direction d_k can be decomposed into $d_k = P_k d_k + d'_k$ with

$$\|d'_k\| \leq \bar{C}(\sum_{j \in I(x^*)} g_j(x_k)^2)^{\frac{1}{2}} + o(\|d^0_k\|^2)$$

for k large enough, for some $\bar{C} > 0$.

Proof.

In view of Proposition 3.5 *iii)*, for k large enough, direction d^0_k satisfies

$$R_k^T d_k^0 = -h_k$$

where h_k is an $|I(x^*)|$-vector whose components are the values $g_j(x_k)$ for $j \in I(x^*)$. One can thus rewrite d^0_k as

$$d^0_k = P_k d^0_k + d^{0\prime}_k$$

with

$$d^{0\prime}_k = -R_k(R_k^T R_k)^{-1} h_k.$$

This implies that, for k large enough,

$$\|d^{0\prime}_k\| \ \leq \ \bar{C}(\sum_{j \in I(x^*)} g_j(x_k)^2)^{\frac{1}{2}}$$

for some $\bar{C}$. The claim then follows from the fact that, in view of the definition of d_k, the properties of $\rho(\cdot)$ and Proposition 3.5 *i)*, we have $d_k = d^0_k + o(\|d^0_k\|^2)$. □

Proof of Proposition 3.6.

From the definition of $\tilde{d}(\cdot,\cdot,\cdot)$ and Step 1 - iii of Algorithm 2.1, it follows that, if $\tilde{d}_k \neq 0$, then $D_k := d_k + \tilde{d}_k$ is solution of the quadratic program.

$$\min_D \frac{1}{2}\langle D, HD\rangle + \langle \nabla f(x_k), D\rangle$$

$$\text{s.t. } g_j(x_k) + \langle \nabla g_j(x_k), D\rangle \leq -\|d_k\|^\tau + g_j(x_k) + \langle \nabla g_j(x_k), d_k\rangle - g_j(x_k + d_k), \quad j = 1, \ldots\ m.$$

From the second order expansion about x_k of $g_j(x_k + d_k)$, $j = 1, \ldots m$, it can be seen that D_k is solution of the same quadratic program as the one for d_k^0 with right hand side perturbed by $O(\|d_k\|^2)$. Thus in view of Proposition 3.5-iii, for k large enough, D_k satisfies, together with some multiplier vector μ^D, the condition

$$H_k D_k + \nabla f(x_k) + \sum_{j \in I(x^*)} \mu_j^D \nabla g_j(x_k) = 0$$

$$g_j(x_k) + \langle g_j(x_k), D_k \rangle = O(\|d_k\|^2) \quad \forall j \in I(x^*).$$

Since d_k^0, with some multiplier μ, satisfies that same equations but with 0 in the right hand side and since, in view of (3.1), (3.4), Assumption A3 and the second order sufficiency condition of optimality on x^*, the linear system is uniformly invertible, it follows that

$$D_k = d_k^0 + O(\|d_k\|^2).$$

Since, from the definition of d_k and ρ_k and the properties of $d^1(\cdot)$ we have

$$d_k = d_k^0 + o(\|d_k^0\|^2) \ ,$$

if follows that

$$\tilde{d}_k = O(\|d_k\|^2) \tag{A.9}$$

The considerations above also imply that, for k large enough, the set of active constraints for (2.6) is $I(x^*)$, and thus

$$g_j(x_k + d_k) + \langle \nabla g_j(x_k), \tilde{d}_k \rangle = -\|d_k\|^\tau, \quad j \in I(x^*) \ . \tag{A.10}$$

With (A.9) and (A.10) established, since $\tau \in (2,3)$, the claim can be proved identically to Proposition 4.8 in [10], using Lemmas A1 and A2 and Proposition 3.5 using the fact that, since $\nu \geq 2$ and both d_k^0 and d_k^1 go to 0,

$$d_k = d_k^0 + (d_k^1 - d_k^0) O(\|d_k^0\|^\nu) = d_k^0 + O(\|d_k^0\|^2).$$

Acknowledgements. The authors wish to thank J.L. Zhou for running all the numerical experiments.

References

No	Code	NF	NG	NIT	FV	KKT	EPS	VC
12	VF02AD	11	11	9	−.300000000E+02	.30E-06	.10E-05	.0
	A	7	18	7	−.300000000E+02	.17E-06	.10E-05	.0
	FSQP	7	14	7	−.300000000E+02	.72E-06	.10E-05	.0
29	VF02AD	12	12	11	−.226274177E+02	.56E-05	.10E-04	.19E-09
	A	14	24	10	−.226274170E+02	.13E-06	.10E-04	.0
	FSQP	11	20	10	−.226274170E+02	.41E-05	.10E-04	.0
30	VF02AD	13	13	13	.100000000E+01	.56E-07	.10E-06	.0
	A	14	44	13	.100000000E+01	.21E-07	.10E-06	.0
	FSQP	13	25	13	.100000000E+01	.26E-07	.10E-06	.0
31	VF02AD	9	9	7	.600000000E+01	.14E-07	.10E-04	.73E-11
	A	11	83	8	.600000000E+01	.59E-06	.10E-04	.0
	FSQP	10	21	8	.600000000E+01	.26E-05	.10E-04	.0
33	VF02AD	5	10	5	−.400000000E+01	.42E-13	.10E-07	.0
	A	4	17	4	−.400000000E+01	.51E-12	.10E-07	.0
	FSQP	4	11	4	−.400000000E+01	.13E-11	.10E-07	.0
34	VF02AD	8	16	8	−.834032443E+00	.12E-08	.10E-07	.12E-08
	A	9	57	8	−.834032445E+00	.18E-12	.10E-07	.0
	FSQP	7	28	7	−.834032443E+00	.19E-08	.10E-07	.0
43	VF02AD	13	39	9	−.440000000E+02	.79E-06	.10E-04	.15E-08
	A	10	67	10	−.440000000E+02	.11E-06	.10E-04	.0
	FSQP	11	51	9	−.440000000E+02	.12E-05	.10E-04	.0
57	VF02AD	4	4	2	.306463060E−01	.26E-05	.10E-04	.0
	A	17	46	3	.306463061E−01	.26E-06	.10E-04	.0
	FSQP	7	5	3	.306463061E−01	.29E-05	.10E-04	.0
66	VF02AD	7	14	7	.518163273E+00	.28E-08	.10E-07	.57E-11
	A	8	30	8	.518163274E+00	.12E-08	.10E-07	.0
	FSQP	8	30	8	.518163274E+00	.50E-09	.10E-07	.0
67	VF02AD	36	504	26	−.120892095E+03	.27E-06	.10E-04	.0
	A			#				
	FSQP	21	305	21	−.116211927E+04	.52E-05	.10E-04	.0
70	VF02AD	35	35	33	.940262492E−02	.29E-07	.10E-06	.0
	A	34	69	30	.940197325E−02	.26E-07	.10E-06	.0
	FSQP	31	35	29	.940197325E−02	.13E-07	.10E-06	.0
84	VF02AD	9	54	5	−.528033513E+07	.36E-02	.10E-01	.35E-05
	A	3	28	3	−.528033512E+07	.18E-02	.10E-01	.0
	FSQP	4	30	4	−.528033513E+07	.66E-09	.10E-01	.0
85	VF02AD	46	1748	39	−.239083247E+01	.60E-03	.10E-02	.0
	A			*				
	FSQP	35	1458	34	−.239964896E+01	.40E-03	.10E-02	.0
93	VF02AD	12	24	10	.135075963E+03	.91E-03	.10E-02	.51E-08
	A			*				
	FSQP	15	58	12	.135075968E+03	.37E-03	.10E-02	.0
100	VF02AD	20	80	13	.680630057E+03	.27E-04	.10E-03	.58E-10
	A	43	240	15	.680630057E+03	.68E-05	.10E-03	.0
	FSQP	23	114	16	.680630057E+03	.62E-06	.10E-03	.0
113	VF02AD	15	120	12	.243065532E+02	.27E-03	.10E-02	.78E-08
	A	15	324	14	.243062091E+02	.54E-05	.10E-02	.0
	FSQP	12	108	12	.243063805E+02	.81E-03	.10E-02	.0
117	VF02AD	16	89	16	.323486790E+02	.45E-04	.10E-03	.45E-04
	A	28	741	22	.323486792E+02	.74E-05	.10E-03	.0
	FSQP	20	219	19	.323486790E+02	.56E-04	.10E-03	.0

Table 1

[1] "Harwell subroutine library", Library Reference Manual (Harwell, England, 1985).

[2] P.T. Boggs and J.W. Tolle, "A strategy for global convergence in a sequential quadratic programming algorithm", SIAM Journal on Numerical Analysis 26(1989)600-623.

[3] G. DiPillo and L. Grippo, "A continuously differentiable exact penalty function for nonlinear programming with inequality constraints", SIAM Journal on Control and Optimization 23(1985)72-84.

[4] M.K.H. Fan, L.-S. Wang, J. Koninckx and A.L. Tits, "Software package for optimization-based design with user-supplied simulators", IEEE Control System Magazine 9(1989).

[5] R. Fletcher, "Numerical experiments with an exact L_1 penalty function method", in: Nonlinear programming 4, O.L. Mangasarian, R.R. Meyer and S.M. Robinson, eds., (Academic Press, New York, 1981) pp. 99-129.

[6] P.E. Gill, W. Murray, M.A. Saunders and M.H. Wright, "User's guide for QPSOL (version 3.2): A Fortran package for quadratic programming", Systems Optimization Laboratory, Technical Report SOL 84-6, Stanford University (Stanford, CA, 1984).

[7] S.P. Han, "A globally convergent method for nonlinear programming", Journal of Optimization Theory and Applications 22(1977) 297-309.

[8] W. Hock and K. Schittkowski, Test examples for nonlinear programming codes, Lecture Notes in Economics and Mathematical Systems (187) (Springer Verlag, 1981).

[9] D. Q. Mayne and E. Polak, "A superlinearly convergent algorithm for constrained optimization problems", Mathematical Programming Studies 16(1982)45-61.

[10] E.R. Panier and A.L. Tits, "A superlinearly convergent feasible method for the solution of inequality constrained optimization problems", SIAM Journal on Control and Optimization 25(1987)934-950.

[11] E.R. Panier, A.L. Tits and J.N. Herskovits, "A QP-free, globally convergent locally superlinearly convergent algorithm for inequality constrained optimization", SIAM Journal on Control and Optimization 26(1988)788-811.

[12] M.J.D. Powell, "A fast algorithm for nonlinearly constrained optimization calculation", in: Numerical Analysis, Dundee, 1977, Lecture Notes in Mathematics 630, G.A. Watson, ed., (Springer-Verlag, 1978) pp. 144-157.

[13] 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, 1978) pp. 27-63.

[14] M.J.D. Powell and Y.X. Yuan, "A recursive quadratic programming algorithm that uses differentiable exact penalty functions", Mathematical Programming 35(1986)265-278.

[15] S.M. Robinson, "Perturbed Kuhn-Tucker Points and rates of convergence for a class of nonlinear-programming algorithms", Mathematical Programming 9(1974)1-16.

[16] J. Stoer, "The convergence of sequential quadratic programming methods for solving nonlinear programs", in: Recent Advances in Communication and Control Theory, R.E. Kalman, G.I. Marchuk, A.E. Ruberti and A.J. Viterbi, eds., (Optimization Software, Inc., New York, N.Y., 1987) pp. 412-421.

[17] J. Zhou and A.L. Tits, "User's guide for FSQP version 2.3. A Fortran code for solving optimization problems, possibly minimax, with general inequality constraints and linear equality constraints, generating feasible iterates". SRC TR-90-60r1b, Systems Research Center, University of Maryland, (College Park, MD, 1991).