Open access • Journal Article • DOI:10.1093/IMANUM/DRQ037

A sequential quadratic programming algorithm with an additional equality constrained phase — Source link

José Luis Morales, Jorge Nocedal, Yuchen Wu

Institutions: Instituto Tecnológico Autónomo de México, Northwestern University

Published on: 01 Apr 2012 - Ima Journal of Numerical Analysis (Oxford University Press)

Topics: Sequential quadratic programming, Quadratic programming, Quadratically constrained quadratic program, Second-order cone programming and Nonlinear programming

A Sequential Quadratic Programming Algorithm with an Additional Equality Constrained Phase

José Luis Morales* Jorge Nocedal † Yuchen Wu†

December 28, 2008

Abstract

A sequential quadratic programming (SQP) method is presented that aims to overcome some of the drawbacks of contemporary SQP methods. It avoids the difficulties associated with indefinite quadratic programming subproblems by defining this subproblem to be always convex. The novel feature of the approach is the addition of an equality constrained phase that promotes fast convergence and improves performance in the presence of ill conditioning. This equality constrained phase uses exact second order information and can be implemented using either a direct solve or an iterative method. The paper studies the global and local convergence properties of the new algorithm and presents a set of numerical experiments to illustrate its practical performance.

1 Introduction

Sequential quadratic programming (SQP) methods are very effective techniques for solving small, medium-size and certain classes of large-scale nonlinear programming problems. They are often preferable to interior-point methods when a sequence of related problems must be solved (as in branch and bound methods) and more generally, when a good estimate of the solution is available. Some SQP methods employ convex quadratic programming subproblems for the step computation (typically using quasi-Newton Hessian approximations) while other variants define the Hessian of the SQP model using second derivative information, which can lead to nonconvex quadratic subproblems; see [27, 1] for surveys on SQP methods.

*Departamento de Matemáticas, Instituto Tecnológico Autónomo de México, México. This author was supported by Asociación Mexicana de Cultura AC and CONACyT-NSF grant J110.388/2006.

†Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL, 60208-3118, USA. These authors was supported by Department of Energy grant DE-FG02-87ER25047 and National Science Foundation grant CCF-0514772.

All these approaches have drawbacks. SQP methods that employ convex quasi-Newton approximations [36, 22, 16] can be slow when solving large or badly scaled problems, whereas methods that employ the exact Hessian of the Lagrangian [19] are confronted with the difficult task of computing global solutions of indefinite quadratic programs [27]. In this paper, we propose an algorithm that aims to overcome some of these drawbacks by formulating the SQP iteration in two phases. The algorithm first solves a convex quadratic program to estimate the optimal active set, and then employs second-derivative information in an additional equality constrained (EQP) phase that promotes rapid convergence. The EQP subproblem is normally less expensive to solve than a general quadratic program and allows for the application of iterative methods that scale up well in the number of variables. We call our algorithm SQP+ because of the incorporation of the additional EQP phase.

Recently, Gould and Robinson [28] proposed an SQP algorithm that has some similarities with the method discussed here. Their method solves a convex quadratic programming problem to compute a so-called predictor step. But rather than using this step to estimate a working set as is done in our algorithm, they then solve another inequality constrained quadratic programming problem (now using second derivative information) and employ the predictor step to guide the solution of this second quadratic program. Other methods that separate the SQP approach in two phases are discussed in [17].

The nonlinear programming problem under consideration is stated as

$$\begin{aligned} \min_{x} \quad & f(x) \\ \text{s.t.} \quad & h(x) = 0, \\ & g(x) \geq 0, \end{aligned} \tag{1.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$, $h : \mathbb{R}^n \to \mathbb{R}^m$ and $g : \mathbb{R}^n \to \mathbb{R}^t$ are smooth functions. We write the Lagrangian of this problem as

$$L(x, \lambda, \mu) = f(x) - \lambda^T h(x) - \mu^T g(x). \tag{1.2}$$

The SQP approach presented in this paper can be implemented both in a line search or a trust region framework. In either case, global convergence can be promoted by imposing decrease in the ℓ_1 merit function

$$\phi_\pi(x) = f(x) + \pi \|h(x)\|_1 + \pi \|g(x)^-\|_1, \tag{1.3}$$

where $g(x)^- \triangleq \max\{0,\ -g_i(x)\}$ and $\pi > 0$ is a penalty parameter.

The paper is organized in 6 sections. In Section 2 we describe the proposed SQP method and outline various implementation options. In Sections 3 and 4 we discuss the global and local convergence properties of the new algorithm, and in Section 5 we report the results of some numerical experiments. Some concluding remarks are made in Section 6.

2 The SQP+ Method

The SQP+ approach can be implemented in a line search or trust region framework. For concreteness, we consider only a line search implementation in this paper. Given a primal-dual iterate (x_k, λ_k, μ_k), the algorithm first computes a step d_k^{IQ} in the primal variables x

by solving the standard quadratic programming subproblem

$$\min_d \quad \nabla f(x_k)^T d + d^T B_k d \tag{2.1a}$$

$$\text{subject to} \quad h(x_k) + H(x_k)d = 0, \tag{2.1b}$$

$$g(x_k) + G(x_k)d \geq 0, \tag{2.1c}$$

where H and G are the Jacobian matrices of h and g, respectively, and B_k is a *positive definite* approximation to the Hessian of the Lagrangian $\nabla^2_{xx} L(x_k, \lambda_k, \mu_k)$. In practice, we would like for B_k to be as simple as possible to keep the cost of solving the convex quadratic program (2.1) to a minimum. The multipliers associated with the solution of (2.1) are denoted by $\lambda^{\text{IQ}}_{k+1}, \mu^{\text{IQ}}_{k+1}$ and will be referred to as the "IQP multipliers". One of the goals of this *IQP phase*, is to provide a good estimate, say $\mathcal{W}_k$, of the optimal active set. To this end, we solve problem (2.1) accurately and define $\mathcal{W}_k$ as the indices of the constraints (2.1*b*)-(2.1*c*) that are satisfied as equalities at the solution d^{IQ}_k, i.e.,

$$\mathcal{W}_k = \{i \in \mathcal{E}\} \cup \{i \in \mathcal{I} \mid g_i(x_k) + \nabla g_i(x_k)^T d^{\text{IQ}}_k = 0\}, \tag{2.2}$$

where $\mathcal{E}, \mathcal{I}$ are the index sets of the equality and inequality constraints, respectively.

To compensate for the limitations of the IQP phase and to try to promote a fast rate of convergence, the algorithm computes an additional step by solving an equality constrained quadratic problem (EQP) in which the constraints in $\mathcal{W}_k$ are imposed as equalities and all other constraints are (temporarily) ignored. The EQP subproblem is given by

$$\min_d \quad (\nabla f(x_k) + W_k d^{\text{IQ}}_k)^T d + \tfrac{1}{2} d^T W_k d \tag{2.3a}$$

$$\text{subject to} \quad \nabla h_i(x_k)^T d = 0, \quad i \in \mathcal{E}, \tag{2.3b}$$

$$\nabla g_i(x_k)^T d = 0, \quad i \in \mathcal{I} \cap \mathcal{W}_k, \tag{2.3c}$$

where

$$W_k \triangleq \nabla^2_{xx} L(x_k, \lambda_k, \mu_k) \tag{2.4}$$

and $\nabla h_i(x_k)^T$ and $\nabla g_i(x_k)^T$ denote the rows of the matrices $H(x_k)$ and $G(x_k)$, respectively. Problem (2.3) could be unbounded because, away from the solution, the Hessian of the Lagrangian may not be positive definite on the tangent space of the constraints defined by $\mathcal{W}_k$. In this case, we can add a regularization term to the objective (2.3*a*) or include a trust region constraint in problem (2.3) to ensure that the EQP subproblem is well defined.

We denote an approximate solution to (2.3) by d^{EQ}_k and the corresponding Lagrange multipliers by $\lambda^{\text{EQ}}_{k+1}, \mu^{\text{EQ}}_{k+1}$. In Section 3 we show that there is much freedom in the selection of the EQP step. There are two effective procedures for computing d^{EQ}_k, namely a projected conjugate gradient method [29, 12, 25] and the direct solution of the (possibly modified) KKT system of (2.3) [37, 2].

Next, the algorithm computes a contraction parameter $\beta \in [0, 1]$ such that the step $d = d^{\text{IQ}}_k + \beta d^{\text{EQ}}_k$ satisfies all linearized constraints that are not in the working set $\mathcal{W}_k$, i.e.,

$$g_i(x_k) + \nabla g_i(x_k)^T d \geq 0, \quad i \in \mathcal{W}^c_k, \tag{2.5}$$

where $\mathcal{W}_k^c$ denotes the complement of $\mathcal{W}_k$. Thus, all the steps of the algorithm will satisfy the linearized constraints (2.1b)-(2.1c).

Before the line search is performed, the algorithm updates the penalty parameter π in (1.3). For this purpose, we define a model of the merit function ϕ_π around the point x_k as

$$q_\pi(d) = f_k + \nabla f_k^T d + \tfrac{1}{2} d^T B_k d + \pi \|h_k + H_k d\|_1 + \pi \|[g_k + G_k d]^-\|_1, \tag{2.6}$$

where f_k is shorthand for $f(x_k)$ and similarly for other functions. We define the model reduction from x_k to $x_k + d$, by

$$qred(d) = q_\pi(0) - q_\pi(d). \tag{2.7}$$

For a step that satisfies the linearized constraints (2.1b)-(2.1c), we have

$$qred(d) = -\nabla f_k^T d - \tfrac{1}{2} d^T B_k d + \pi(\|h_k\|_1 + \|g_k^-\|_1). \tag{2.8}$$

The update of the penalty parameter is based on the IQP step. As is now common [3, 38, 7], we require that the new penalty parameter π_k be large enough that

$$qred(d_k^{\mathrm{IQ}}) \geq \rho \pi_k (\|h_k\|_1 + \|g_k^-\|_1), \tag{2.9}$$

for some prescribed constant $\rho \in (0, 1)$. From (2.8) we see that condition (2.9) is equivalent to the requirement

$$\pi_k \geq \frac{\nabla f_k^T d_k^{\mathrm{IQ}} + \frac{1}{2} d_k^{\mathrm{IQ}T} B_k d_k^{\mathrm{IQ}}}{(1-\rho)(\|h_k\|_1 + \|g_k^-\|_1)} \triangleq \pi_{\mathrm{trial}}. \tag{2.10}$$

We can enforce this condition by updating the penalty parameter at every iteration k by means of the following rule:

$$\pi_k = \begin{cases} \pi_{k-1} & \text{if (2.9) is satisfied} \\ \pi_{\mathrm{trial}} + \pi_b & \text{otherwise,} \end{cases} \tag{2.11}$$

where $\pi_b > 0$ is a given constant and π_{trial} is defined in (2.10).

The algorithm then performs a backtracking line search along the piecewise linear segment that starts at x_k, goes through $x_k + d_k^{\mathrm{IQ}}$ and ends at $x_k + d_k^{\mathrm{IQ}} + \beta d_k^{\mathrm{EQ}}$. The line search first attempts to find a steplength $\alpha_k \in [\alpha_{\min}, 1]$ that satisfies the sufficient decrease condition

$$\phi_{\pi_k}(x_k + d_k^{\mathrm{IQ}} + \alpha_k \beta d_k^{\mathrm{EQ}}) \leq \phi_{\pi_k}(x_k) - \sigma qred(d_k^{\mathrm{IQ}}), \tag{2.12}$$

for some given constant $\sigma \in (0, 1)$, and where $\alpha_{\min}$ is a threshold such as 0.25. If this line search is successful, we define the total step as

$$d_k = d_k^{\mathrm{IQ}} + \alpha_k \beta d_k^{\mathrm{EQ}}. \tag{2.13}$$

Otherwise, we choose a constant $\tau \in (0, 1)$ and let $\alpha_k \in (0, 1]$ be the first member of the sequence $\{1, \tau, \tau^2, \ldots\}$ such that

$$\phi_{\pi_k}(x_k + \alpha_k d_k^{\mathrm{IQ}}) \leq \phi_{\pi_k}(x_k) - \sigma \alpha_k qred(d_k^{\mathrm{IQ}}), \tag{2.14}$$

where σ is the same constant as in (2.12); we then set

$$d_k = \alpha_k d_k^{\mathrm{IQ}}. \tag{2.15}$$

Regardless of whether d_k is defined by (2.13) or (2.15), the new primal iterate is defined as

$$x_{k+1} = x_k + d_k. \tag{2.16}$$

The new Lagrange multipliers are defined at the EQP point. First, we set the multipliers corresponding to the inactive linearized constraints to zero. The rest of the multipliers are set to the EQP multipliers $\lambda_{k+1}^{\mathrm{EQ}}$, μ_{k+1}^{EQ} — except that if any such multipliers are negative, they are set to zero.

Before describing the algorithm in more detail, we introduce the following notation to denote subvectors. Given a vector, say μ, and a working set $\mathcal{W}_k$, we denote by $\mu_{\mathcal{W}_k}$ the subvector of μ restricted to the set $\mathcal{W}_k$, i.e.,

$$\mu_{\mathcal{W}_k} = [\mu_i]_{i \in \mathcal{W}_k}.$$

The new SQP algorithm is specified as follows.

Algorithm I

Initial data: $x_0, \pi_0 > 0$, $\pi_b > 0$, $\rho > 0$, $\tau \in (0,1)$ and $\sigma \in (0,1)$.

For $k = 1, 2, \ldots$ until the KKT conditions for the nonlinear program (1.1) are satisfied

1. Define a positive definite matrix B_k and compute the step d_k^{IQ} and multipliers $\lambda_{k+1}^{\mathrm{IQ}}, \mu_{k+1}^{\mathrm{IQ}}$ by solving the subproblem (2.1);
2. Determine the working set $\mathcal{W}_k$;
3. Compute the EQP step d_k^{EQ} and multipliers $\lambda_{k+1}^{\mathrm{EQ}}, \mu_{k+1}^{\mathrm{EQ}}$ by finding an approximate solution of problem (2.3);
4. Compute the largest number $\beta \in [0,1]$ that ensures that the step $d = d_k^{\mathrm{IQ}} + \beta d_k^{\mathrm{EQ}}$ satisfies the constraints (2.5);
5. Compute the penalty parameter π_k by (2.11);
6. Compute the steplength α_k, define d_k by (2.13) or (2.15), and set $x_{k+1} = x_k + d_k$.
7. Set
$$\lambda_{k+1} = \lambda_{k+1}^{\mathrm{EQ}}, \quad [\mu_{k+1}]_{\mathcal{W}_k} = \max\left(0, \mu_{k+1}^{\mathrm{EQ}}\right), \quad [\mu_{k+1}]_{\mathcal{W}_k^c} = 0,$$
where $\mathcal{W}_k^c$ denotes the complement of $\mathcal{W}_k$.

We have defined the model q_π in terms of the positive definite Hessian approximation B_k so that the IQP step drives the global convergence of the algorithm — while the EQP phase yields superlinear convergence.

Many enhancements and safeguards are needed to transform this general (and somewhat idealized) algorithm into a practical approach. For example, the constraints in (2.1) are not always feasible. This issue can be resolved by modifying the constraints using relaxations [32, 3] or by following a penalty approach [18, 10, 6]. The algorithm should also include regularization strategies for controlling singularity and ill conditioning [24, 13], as well as second-order corrections [18] or watchdog steps [9] to improve the efficiency of the iteration. Step 4 is quite restrictive (for simplicity) and in a practical implementation we might allow the EQP step to violate some of the linearized constraints that are not in the working set. In Section 6 we comment on a variant that employs a projection in the EQP phase instead of backtracking to satisfy the constraints (2.5).

A fully developed implementation of the proposed approach is outside the scope of this paper. We focus, instead, on some of the key aspects of the algorithm so that its potential can be assessed in a simple setting. One of the main questions we wish to investigate is whether the EQP phase does, in fact, add significant benefits to the standard SQP approach.

One motivation for the SQP+ algorithm stems from the difficulties that have been encountered in trying to introduce second-derivative information in SQP codes that were originally designed to solve convex quadratic subproblems, such as SNOPT. Another motivation arose from the numerical experience with the sequential linear-quadratic programming (SLQP) method described in [4, 5]. The linear programming phase of that algorithm is normally able to make a good selection of the working set, but it can be erratic in the presence of degenerate constraints. Moreover, it has proved to be difficult to warm start this linear programming phase because the trust region constraint, which is defined as a box constraint [20, 4, 11], is typically active at every iteration and it is difficult to predict which sides of the box will be active. The IQP phase of the SQP+ algorithm is both easier to warm start and provides a better working set estimation than the SLQP method — at the price of a more expensive step computation.

3 Global Convergence Properties

In this section we show that Algorithm I enjoys favorable global convergence properties. We make the following assumptions about problem (1.1) and the iterates generated by the algorithm.

Assumptions 3.1

(a) The sequence $\{x_k\}$ generated by Algorithm I is contained in a convex set Ω where the functions f, h, g are twice differentiable; the sequence $\{f_k\}$ is bounded below and the sequences $\{\nabla f_k\}$, $\{h_k,\}$, $\{g_k\}$, $\{H_k\}$ and $\{G_k\}$ are bounded.

(b) The quadratic program (2.1) *is feasible for all k;*

(c) The sequence of IQP multipliers is bounded, i.e. there exists a positive constant η *such that* $\|(\lambda_k^{IQ}, \mu_k^{IQ})\|_\infty \leq \eta$ *for all* k*;*

(d) The sequence of Hessian approximations $\{B_k\}$ *is bounded above and there exists a constant* $M > 0$ *such that for all* k*,*

$$\tfrac{1}{2} d_k^{IQT} B_k d_k^{IQ} \geq M \|d_k^{IQ}\|_2^2. \tag{3.1}$$

These assumptions are fairly restrictive, in particular (b) and (c). Although they have often been used in the literature (see e.g. Powell [34]), recent studies of SQP methods establish global convergence properties under weaker assumptions. This requires, however, that the SQP method be modified significantly by incorporating constraint relaxations, Hessian modifications, trust regions or other devices [13, 6, 10]. The resulting algorithms are complex and the analysis is long and laborious. By making benign assumptions, we can develop a fairly compact convergence analysis that highlights the key properties of the SQP+ approach.

We begin the analysis by noting that the primal-dual solution $(d_k^{\rm IQ}, \lambda_{k+1}^{\rm IQ}, \mu_{k+1}^{\rm IQ})$ of the IQP subproblem (2.1) satisfies the following first-order conditions:

$$\nabla f_k + B_k d_k^{\rm IQ} - {H_k}^T \lambda_{k+1}^{\rm IQ} - {G_k}^T \mu_{k+1}^{\rm IQ} = 0 \tag{3.2a}$$

$$h_k + H_k d_k^{\rm IQ} = 0 \tag{3.2b}$$

$$g_k + G_k d_k^{\rm IQ} \geq 0 \tag{3.2c}$$

$$\mu_{k+1}^{\rm IQ} \geq 0 \tag{3.2d}$$

$${\mu_{k+1}^{\rm IQ}}^T (g_k + G_k d_k^{\rm IQ}) = 0. \tag{3.2e}$$

It is not difficult to show (cf. [31, page 628]) that the directional derivative of the merit function ϕ_{π_k} at a point x_k along the direction $d_k^{\rm IQ}$ satisfies

$$D\phi_{\pi_k}(x_k; d_k^{\rm IQ}) \leq \nabla f_k^T d_k^{\rm IQ} - \pi \|h(x_k)\|_1 - \pi \|g(x_k)^-\|_1.$$

By comparing the right hand side of this expression with (2.8), we obtain

$$D\phi_{\pi_k}(x_k; d_k^{\rm IQ}) \leq -qred(d_k^{\rm IQ}) - \tfrac{1}{2} d_k^{\rm IQT} B_k d_k^{\rm IQ}. \tag{3.3}$$

Recalling from (2.9) that $qred(d_k^{\rm IQ}) \geq 0$ and since B_k is positive definite, we conclude that $d_k^{\rm IQ}$ is a descent direction for ϕ_{π_k}. Relation (3.3) also shows that, by using $qred(d_k^{\rm IQ})$ in the right hand sides of (2.12) and (2.14) (instead of using the directional derivative $D\phi_{\pi_k}(x_k; d_k^{\rm IQ})$ as in a standard Armijo condition), the sufficient decrease condition is less demanding and accounts for the nondifferentiability of the penalty function. This allows the algorithm to take longer steps.

An immediate consequence of Assumption 3.1-(c) is that the penalty parameters π_k generated by the update rule (2.11) are bounded.

Lemma 3.2 *There is an index* $\bar{k}$ *and a positive constant* $\bar{\pi}$ *such that* $\pi_k = \bar{\pi}$ *for all* $k \geq \bar{k}$.

Proof. From (3.2*b*) and (3.2*e*) we have that

$$(H_k d_k^{\rm IQ})^T \lambda_{k+1}^{\rm IQ} = -h_k^T \lambda_{k+1}^{\rm IQ}, \qquad (G_k d_k^{\rm IQ})^T \mu_{k+1}^{\rm IQ} = -g_k^T \mu_{k+1}^{\rm IQ}. \tag{3.4}$$

Next, since $\mu_{k+1}^{\rm IQ} \geq 0$, we have that

$$-(\mu_{k+1}^{\rm IQ})^T g_k \leq (\mu_{k+1}^{\rm IQ})^T \max\{0, -g_k\} = (\mu_{k+1}^{\rm IQ})^T g_k^- \leq \|\mu_{k+1}^{\rm IQ}\|_\infty \|g_k^-\|_1,$$

and we also have

$$(\lambda_{k+1}^{\rm IQ})^T h_k \leq \|\lambda_{k+1}^{\rm IQ}\|_\infty \|h_k\|_1.$$

By combining these two relations with (3.2*a*) and (3.4), and by Assumptions 3.1-(c), we obtain

$$\begin{aligned} \nabla f_k^T d_k^{\rm IQ} + \tfrac{1}{2} d_k^{\rm IQ T} B_k d_k^{\rm IQ} &\leq \nabla f_k^T d_k^{\rm IQ} + d_k^{\rm IQ T} B_k d_k^{\rm IQ} \\ &\leq \|(\lambda_{k+1}^{\rm IQ}, \mu_{k+1}^{\rm IQ})\|_\infty (\|h_k\|_1 + \|g_k^-\|_1) \\ &\leq \eta(\|h_k\|_1 + \|g_k^-\|_1). \end{aligned}$$

From this relation and (2.10) we have that $\pi_{\rm trial} \leq \eta/(1-\rho)$. Consequently, we have from (2.11) that for all k,

$$\pi_k \leq \max\left\{\pi_0, \frac{\eta}{1-\rho} + \pi_b\right\},$$

showing that the sequence of penalty parameters is bounded from above. Finally, note that when the penalty parameter is increased, it is increased by the finite amount π_b. This implies that the sequence of penalty parameters is eventually constant, which completes the proof. □

The next result is crucial.

Lemma 3.3 *Under Assumptions 3.1, Algorithm I yields the limit*

$$\lim_{k\to\infty} qred(d_k^{IQ}) = 0. \tag{3.5}$$

Proof. By Lemma 3.2, there exists an integer $\bar{k}$ such that for all $k \geq \bar{k}$, $\pi_k = \bar{\pi}$. Since for the purpose of proving convergence, the initial finite number of iterations are not relevant, we consider only those iterations with $k \geq \bar{k} + 1$.

Let $\mathcal{T}_e$ denote the set of iterates for which the line search along the EQP direction was successful, and hence (2.12) is satisfied. In this case we have

$$\phi_{\bar{\pi}}(x_k) - \phi_{\bar{\pi}}(x_{k+1}) \geq \sigma qred(d_k^{\rm IQ}) \ \text{ for } k \in \mathcal{T}_e. \tag{3.6}$$

Similarly, let $\mathcal{T}_i$ denote the set of iterates for which a backtracking line search along the direction $d^{\rm IQ}$ is performed. In this case we have from (2.14) that

$$\phi_{\bar{\pi}}(x_k) - \phi_{\bar{\pi}}(x_{k+1}) \geq \sigma \alpha_k qred(d_k^{\rm IQ}) \ \text{ for } k \in \mathcal{T}_i. \tag{3.7}$$

By (2.9) we have that $qred(d_k^{\rm IQ}) \geq 0$ for all k and therefore the sequence $\{\phi_{\bar{\pi}}(x_{k+1})\}$ is monotonically decreasing. By Assumption 3.1-(a), $f(x_k)$ is bounded from below and $g(x_k)$, $h(x_k)$ are bounded, and so is $\phi_{\bar{\pi}}(x_k)$. Thus, $\{\phi_{\bar{\pi}}(x_k)\}$ must converge, i.e.,

$$\lim_{k\to\infty} \left(\phi_{\bar{\pi}}(x_{k+1}) - \phi_{\bar{\pi}}(x_k)\right) = 0,$$

which implies

$$\lim_{k\in\mathcal{T}_e,k\to\infty} qred(d_k^{\rm IQ}) = 0 \quad \text{and} \quad \lim_{k\in\mathcal{T}_i,k\to\infty} \alpha_k qred(d_k^{\rm IQ}) = 0. \tag{3.8}$$

If the set $\mathcal{T}_i$ is finite, the limit (3.5) holds since $k \in \mathcal{T}_e$ for all large k. Therefore, we assume that $\mathcal{T}_i$ is infinite and show that Assumptions 3.1, the backtracking line search procedure and the second limit in (3.8) imply that $\lim_{k\in\mathcal{T}_i,k\to\infty} qred(d_k^{\rm IQ}) = 0$; this will prove the lemma.

Recalling the definitions (1.3) and (2.6) and Assumptions 3.1-(a) we have that

$$\begin{aligned}\phi_{\bar\pi}(x_k + \alpha_k d_k^{\rm IQ}) - q_{\bar\pi}(\alpha_k d_k^{\rm IQ}) \le\ & f(x_k + \alpha_k d_k^{\rm IQ}) - (f_k + \alpha_k \nabla f_k^T d_k^{\rm IQ}) \\ & + \bar\pi(\|h(x_k + \alpha_k d_k^{\rm IQ})\|_1 - \|h_k + \alpha_k H_k d_k^{\rm IQ}\|_1) \\ & + \bar\pi(\|g(x_k + \alpha_k d_k^{\rm IQ})^-\|_1 - \|[g_k + \alpha_k G_k d_k^{\rm IQ}]^-\|_1) \\ \le\ & L_1 \alpha_k^2 \|d_k^{\rm IQ}\|_2^2,\end{aligned} \tag{3.9}$$

for some constant $L_1 > 0$ independent of k. From (3.2a) we have

$$d_k^{\rm IQ} = B_k^{-1}[-\nabla f_k + {H_k}^T \lambda_{k+1}^{\rm IQ} + {G_k}^T \mu_{k+1}^{\rm IQ}].$$

By Assumptions 3.1-(a), all the terms inside the square brackets are bounded and the matrices B_k are uniformly positive definite. Therefore, for all k, there is a constant L_2 such that

$$\|d_k^{\rm IQ}\|_2^2 \le L_2,$$

and (3.9) yields

$$\phi_{\bar\pi}(x_k + \alpha_k d_k^{\rm IQ}) - q_{\bar\pi}(\alpha_k d_k^{\rm IQ}) \le L_1 L_2 \alpha_k^2. \tag{3.10}$$

The positive definiteness of B_k also implies that $q_{\bar\pi}$ is a convex function, and therefore,

$$q_{\bar\pi}(\alpha_k d_k^{\rm IQ}) \le (1 - \alpha_k) q_{\bar\pi}(0) + \alpha_k q_{\bar\pi}(d_k^{\rm IQ}).$$

Recalling (2.7), this inequality gives

$$q_{\bar\pi}(\alpha_k d_k^{\rm IQ}) - q_{\bar\pi}(0) \le -\alpha_k qred(d_k^{\rm IQ}).$$

Combining this expression with (3.10), and noting that $\phi_{\bar\pi}(x_k) = q_{\bar\pi}(0)$, we obtain

$$\begin{aligned}\phi_{\bar\pi}(x_k + \alpha_k d_k^{\rm IQ}) - \phi_{\bar\pi}(x_k) &= \left[\phi_{\bar\pi}(x_k + \alpha_k d_k^{\rm IQ}) - q_{\bar\pi}(\alpha_k d_k^{\rm IQ})\right] + \left[q_{\bar\pi}(\alpha_k d_k^{\rm IQ}) - \phi_{\bar\pi}(x_k)\right] \\ &\le -\alpha_k qred(d_k^{\rm IQ}) + L_1 L_2 \alpha_k^2.\end{aligned} \tag{3.11}$$

Now, if

$$\alpha_k \le \frac{(1-\sigma)}{L_1 L_2} qred(d_k^{\rm IQ}), \tag{3.12}$$

we have

$$\begin{aligned}\phi_{\bar\pi}(x_k + \alpha_k d_k^{\rm IQ}) - \phi_{\bar\pi}(x_k) &\le -\sigma\alpha_k qred(d_k^{\rm IQ}) - \alpha_k(1-\sigma) qred(d_k^{\rm IQ}) + L_1 L_2 \alpha_k^2 \\ &\le -\sigma\alpha_k qred(d_k^{\rm IQ}),\end{aligned}$$

and therefore, the sufficient decrease condition (2.14) is satisfied. Since $k \in \mathcal{T}_i$, α_k is chosen by a backtracking line search with a contraction factor τ, we conclude that

$$\alpha_k \geq \tau \frac{(1-\sigma)}{L_1 L_2} qred(d_k^{\mathrm{IQ}}).$$

Substituting this inequality in the second limit in (3.8) gives

$$0 = \lim_{k \in \mathcal{T}_i, k \to \infty} \alpha_k qred(d_k^{\mathrm{IQ}}) \geq \lim_{k \in \mathcal{T}_i, k \to \infty} \tau \tfrac{(1-\sigma)}{L_1 L_2} qred^2(d_k^{\mathrm{IQ}}) \geq 0,$$

which implies that $\lim_{k \in \mathcal{T}_i, k \to \infty} qred(d_k^{\mathrm{IQ}}) = 0$. □

We can now establish some limiting properties of the iterates.

Lemma 3.4 *The sequence $\{x_k\}$ generated by Algorithm I is asymptotically feasible, i.e.,*

$$\lim_{k \to \infty} (\|h_k\|_1 + \|g_k^-\|_1) = 0. \tag{3.13}$$

Furthermore,

$$\lim_{k \to \infty} g_k^T \mu_{k+1}^{IQ} = 0 \tag{3.14}$$

and

$$\lim_{k \to \infty} d_k^{IQ} = 0. \tag{3.15}$$

Proof. From Lemma 3.3 and condition (2.9) we get

$$0 = \lim_{k \to \infty} qred(d_k^{\mathrm{IQ}}) \geq \lim_{k \to \infty} \rho \pi_k (\|h_k\|_1 + \|g_k^-\|_1) \geq 0.$$

Since π_k is bounded above by Lemma 3.2, we have that the limit (3.13) holds.

As to (3.14) and (3.15), we first observe from (3.2a) and (3.4) that

$$\begin{aligned} 0 &= \nabla f_k^T d_k^{\mathrm{IQ}} + d_k^{\mathrm{IQ}T} B_k d_k^{\mathrm{IQ}} - (H_k d_k^{\mathrm{IQ}})^T \lambda_{k+1}^{\mathrm{IQ}} - (G_k d_k^{\mathrm{IQ}})^T \mu_{k+1}^{\mathrm{IQ}} \\ &= \nabla f_k^T d_k^{\mathrm{IQ}} + d_k^{\mathrm{IQ}T} B_k d_k^{\mathrm{IQ}} + h_k^T \lambda_{k+1}^{\mathrm{IQ}} + g_k^T \mu_{k+1}^{\mathrm{IQ}} \\ &= \{\nabla f_k^T d_k^{\mathrm{IQ}} + \tfrac{1}{2} d_k^{\mathrm{IQ}T} B_k d_k^{\mathrm{IQ}}\} + \tfrac{1}{2} d_k^{\mathrm{IQ}T} B_k d_k^{\mathrm{IQ}} + h_k^T \lambda_{k+1}^{\mathrm{IQ}} + g_k^T \mu_{k+1}^{\mathrm{IQ}}. \end{aligned} \tag{3.16}$$

Using the definition (2.8) of *qred*, the limits (3.5) and (3.13), and the boundedness of $\{\pi_k\}$, we obtain

$$\lim_{k \to \infty} \nabla f_k^T d_k^{\mathrm{IQ}} + \tfrac{1}{2} d_k^{\mathrm{IQ}T} B_k d_k^{\mathrm{IQ}} = 0, \tag{3.17}$$

and consequently by (3.16)

$$\lim_{k \to \infty} \tfrac{1}{2} d_k^{\mathrm{IQ}T} B_k d_k^{\mathrm{IQ}} + h_k^T \lambda_{k+1}^{\mathrm{IQ}} + g_k^T \mu_{k+1}^{\mathrm{IQ}} = 0. \tag{3.18}$$

By Assumption 3.1-(c), the IQP multipliers are bounded and thus from (3.13) we have

$$\lim_{k \to \infty} h_k^T \lambda_{k+1}^{\mathrm{IQ}} = 0. \tag{3.19}$$

Therefore (3.18) simplifies to

$$\lim_{k\to\infty} \tfrac{1}{2} d_k^{\mathrm{IQ}T} B_k d_k^{\mathrm{IQ}} + g_k^T \mu_{k+1}^{\mathrm{IQ}} = 0. \tag{3.20}$$

We also have from (3.13) that $||g_k^-|| \to 0$, and thus by the boundedness of the IQP multipliers we obtain

$$\lim_{k\to\infty} \sum_{i\in J_k^-} g_i(x_k)[\mu_{k+1}^{\mathrm{IQ}}]_i = 0, \tag{3.21}$$

where $J_k^- = \{i : g_i(x_k) < 0\}$. Therefore,

$$\begin{aligned} \lim_{k\to\infty} g_k^T \mu_{k+1}^{\mathrm{IQ}} &= \lim_{k\to\infty} \left\{ \sum_{i\in J_k^-} g_i(x_k)[\mu_{k+1}^{\mathrm{IQ}}]_i + \sum_{i\in J_k^+} g_i(x_k)[\mu_{k+1}^{\mathrm{IQ}}]_i \right\} \\ &= \lim_{k\to\infty} \sum_{i\in J_k^+} g_i(x_k)[\mu_{k+1}^{\mathrm{IQ}}]_i \\ &\geq 0, \end{aligned} \tag{3.22}$$

where $J_k^+ = \{i : g_i(x_k) > 0\}$. By (3.1) and (3.22) both terms in (3.20) are nonnegative, so we conclude that

$$\lim_{k\to\infty} g_k^T \mu_{k+1}^{\mathrm{IQ}} = 0 \quad \text{and} \quad \lim_{k\to\infty} \tfrac{1}{2} d_k^{\mathrm{IQ}T} B_k d_k^{\mathrm{IQ}} = 0$$

Hence

$$0 = \lim_{k\to\infty} \tfrac{1}{2} d_k^{\mathrm{IQ}T} B_k d_k^{\mathrm{IQ}} \geq \lim_{k\to\infty,} M \|d_k^{\mathrm{IQ}}\|_2^2 \geq 0,$$

which implies that $d_k^{\mathrm{IQ}} \to 0$. □

We can now prove the main result of this section.

Theorem 3.5 *Any limit point of the sequence* $\{x_k, \lambda_{k+1}^{IQ}, \mu_{k+1}^{IQ}\}$ *generated by Algorithm I is a KKT point of the nonlinear program* (1.1).

Proof. The KKT conditions for the nonlinear program (1.1) are given by

$$\nabla f(x) - H(x)^T \lambda - G(x)^T \mu = 0 \tag{3.23a}$$
$$h(x) = 0 \tag{3.23b}$$
$$g(x) \geq 0 \tag{3.23c}$$
$$\mu \geq 0 \tag{3.23d}$$
$$\mu^T g(x) = 0. \tag{3.23e}$$

Consider a limit point $\{x_*, \lambda_*, \mu_*\}$ of the sequence $\{x_k, \lambda_{k+1}^{\mathrm{IQ}}, \mu_{k+1}^{\mathrm{IQ}}\}$. Then there is an index set $\mathcal{K}$ such that $(x_k, \lambda_{k+1}^{\mathrm{IQ}}, \mu_{k+1}^{\mathrm{IQ}})_{k\in\mathcal{K}} \to (x_*, \lambda_*, \mu_*)$. Taking limits in (3.2*a*) we obtain

$$\lim_{k\to\infty, k\in\mathcal{K}} \nabla f_k^T + B_k d_k^{\mathrm{IQ}} - H_k^T \lambda_{k+1}^{\mathrm{IQ}} - G_k^T \mu_{k+1}^{\mathrm{IQ}} = 0.$$

We have shown in Lemma 3.4 that the sequence $\{d_k^{\rm IQ}\}$ converges to zero, and Assumption 3.1-(d) states that the matrices B_k are bounded. Hence,

$$\nabla f(x_*) - H(x_*)^T\lambda_* - G(x_*)^T\mu_* = 0,$$

so that (3.23a) is satisfied. Lemma 3.4 guarantees that x_* is feasible and hence (3.23b) and (3.23c) hold. Condition (3.23e) follows from (3.14), and the nonnegativity condition (3.23d) holds by (3.2d). Consequently, the limit point $\{x_*, \lambda_*, \mu_*\}$ satisfies the KKT conditions (3.23). □

This theorem shows that the primal-dual variables $(x_k, \lambda_{k+1}^{\rm IQ}, \mu_{k+1}^{\rm IQ})$ can be used to terminate the algorithm. An alternative (primal) stop test would be to terminate the iteration when $||d_k^{\rm IQ}||$ is smaller than a given tolerance. One can also base the convergence test on the new iterate, $(x_{k+1}, \lambda_{k+1}, \mu_{k+1})$.

4 Local Convergence Properties

We now show that Algorithm I identifies the optimal active set once an iterate x_k is close enough to a solution satisfying standard conditions. Furthermore, since for large k the EQP phase computes Newton steps along the optimal active set, we show that the rate of convergence of the iteration is quadratic.

We begin by introducing some notation. The working set $\mathcal{W}_k$ is given by (2.2) and the (optimal) active set corresponding to a solution x_* of the nonlinear program (1.1) is defined as

$$\mathcal{W}_* = \{i \in \mathcal{E}\} \cup \{i \in \mathcal{I} \mid g_i(x_*) = 0\}. \tag{4.1}$$

We denote by $c_{\mathcal{W}}$ the vector of constraints in the working set, i.e.,

$$c_{\mathcal{W}}(x_k) = \begin{bmatrix} h(x_k) \\ g_{\mathcal{W}}(x_k) \end{bmatrix}, \tag{4.2}$$

and denote its Jacobian by $A_{\mathcal{W}}$, specifically

$$A_{\mathcal{W}}(x_k) = \begin{bmatrix} H(x_k) \\ G_{\mathcal{W}}(x_k) \end{bmatrix}, \quad \text{with} \quad G_{\mathcal{W}}(x_k) = [\nabla g_i(x_k)^T]_{i\in\mathcal{W}}. \tag{4.3}$$

Our local convergence results are established under the following assumptions.

Assumptions 4.1 *Let (x_*, λ_*, μ_*) be a primal-dual solution to* (1.1) *and let $\mathcal{W}_*$ be the corresponding optimal active set.*

(a) (Smoothness) The functions f, g and h are twice continuously differentiable in a neighborhood of x_*;

(b) (Regularity) The Jacobian of the active constraints at x_*, denoted as $A_{\mathcal{W}_*}(x_*)$, has full row rank;

(c) (Strict Complementarity) $\mu_* + g(x_*) > 0$*;*

(d) (Regularity of IQP Model) The subproblem (2.1) *is feasible for all* k*. The matrices* B_k *are positive definite and their smallest eigenvalue is bounded away from zero;*

(e) (Second Order Sufficiency) The Hessian $W(x, \lambda, \mu)$ *satisfies* $y^T W(x_*, \lambda_*, \mu_*) y > 0$ *for all* $y \neq 0$ *such that* $A_{\mathcal{W}_*}(x_*) y = 0$*.*

Throughout the analysis, we assume that $\|\cdot\|$ denotes the 2-norm, unless indicated otherwise. The following lemma, which was first proved by Robinson [35], states that near a solution x_*, the IQP step identifies the optimal active set $\mathcal{W}_*$. We give a proof of this result because some of the arguments are used again in the proof of Theorem 4.3.

Lemma 4.2 *Suppose that Assumptions 4.1 hold. If* x_k *is sufficiently close to* x_**, the working set* (2.2) *identified by the solution of the IQP subproblem* (2.1) *is the optimal active set, i.e.,* $\mathcal{W}_k = \mathcal{W}_*$*.*

Proof. By Assumption 4.1-(d), the IQP subproblem (2.1) has a unique primal optimal solution d_k^{IQ}, and hence the working set $\mathcal{W}_k$ is uniquely determined by (2.2). Let us recall definitions (4.2) and (4.3), and consider the system

$$\begin{bmatrix} B_k & -A_{\mathcal{W}_*}^T(x_k) \\ A_{\mathcal{W}_*}(x_k) & 0 \end{bmatrix} \begin{bmatrix} d \\ \gamma \end{bmatrix} = - \begin{bmatrix} \nabla f_k \\ c_{\mathcal{W}_*}(x_k) \end{bmatrix}, \tag{4.4}$$

which is defined in terms of the optimal active set $\mathcal{W}_*$. We now show that when x_k is close to x_*, the IQP step can be computed via solution of the system (4.4).

Let us define the neighborhood

$$\mathcal{N}_*(\epsilon) \triangleq \{\ x \mid \|x - x_*\| \leq \epsilon\} \quad \text{with} \quad \epsilon > 0. \tag{4.5}$$

By Assumptions 4.1-(a), (b), (d), for ϵ sufficiently small and $x_k \in \mathcal{N}_*(\epsilon)$, the linear system (4.4) is nonsingular and the inverse of its coefficient matrix is bounded above in norm by some constant $\delta_1 > 0$. Let us define

$$\gamma_* = \begin{bmatrix} \lambda_* \\ [\mu_*]_{\mathcal{W}_*} \end{bmatrix}. \tag{4.6}$$

From (4.4) we have that

$$\begin{bmatrix} d \\ \gamma - \gamma_* \end{bmatrix} = - \begin{bmatrix} B_k & -A_{\mathcal{W}_*}^T(x_k) \\ A_{\mathcal{W}_*}(x_k) & 0 \end{bmatrix}^{-1} \begin{bmatrix} \nabla f_k - A_{\mathcal{W}_*}^T(x_k)\gamma_* \\ c_{\mathcal{W}_*}(x_k) \end{bmatrix}, \tag{4.7}$$

and hence

$$\left\| \begin{bmatrix} d \\ \gamma - \gamma_* \end{bmatrix} \right\| \leq \delta_1 \left\| \begin{bmatrix} \nabla f_k - A_{\mathcal{W}_*}^T(x_k)\gamma_* \\ c_{\mathcal{W}_*}(x_k) \end{bmatrix} \right\|. \tag{4.8}$$

Furthermore, Assumptions 4.1-(a) imply that ∇f, $c_{\mathcal{W}_*}$, $A_{\mathcal{W}_*}$ are Lipschitz continuous and therefore for all $x_k \in \mathcal{N}_*(\epsilon)$ there exist constants κ_f, κ_c and κ_a such that

$$\|\nabla f_k - \nabla f(x_*)\| \leq \kappa_f \epsilon, \quad \|c_{\mathcal{W}_*}(x_k) - c_{\mathcal{W}_*}(x_*)\| \leq \kappa_c \epsilon, \quad \|A_{\mathcal{W}_*}(x_k) - A_{\mathcal{W}_*}(x_*)\| \leq \kappa_a \epsilon. \tag{4.9}$$

By (4.6), we can express the KKT conditions (3.23) of the nonlinear program (1.1) as

$$\begin{bmatrix} \nabla f(x_*) - A_{\mathcal{W}_*}^T(x_*)\gamma_* \\ c_{\mathcal{W}_*}(x_*) \end{bmatrix} = 0, \tag{4.10}$$

together with

$$[\mu_*]_{\mathcal{W}_*} > 0, \quad [\mu_*]_{\mathcal{W}_*^c} = 0, \quad g_{\mathcal{W}_*^c}(x_*) > 0, \tag{4.11}$$

where the second condition follows from Assumption 4.1-(c). Therefore, we have from (4.10) and (4.9) that

$$\begin{aligned} \left\| \begin{bmatrix} \nabla f_k - A_{\mathcal{W}_*}^T(x_k)\gamma_* \\ c_{\mathcal{W}_*}(x_k) \end{bmatrix} \right\| &= \left\| \begin{bmatrix} (\nabla f_k - \nabla f(x_*)) - (A_{\mathcal{W}_*}(x_k) - A_{\mathcal{W}_*}(x_*))^T \gamma_* \\ c_{\mathcal{W}_*}(x_k) - c_{\mathcal{W}_*}(x_*) \end{bmatrix} \right\| \\ &\leq \|\nabla f_k - \nabla f(x_*)\| + \|c_{\mathcal{W}_*}(x_k) - c_{\mathcal{W}_*}(x_*)\| + \|A_{\mathcal{W}_*}(x_k) - A_{\mathcal{W}_*}(x_*)\|\|\gamma_*\| \\ &\leq \big(\kappa_f + \kappa_c + \kappa_a \|\gamma_*\|\big)\epsilon. \end{aligned} \tag{4.12}$$

If we define $\kappa_o = \kappa_f + \kappa_c + \kappa_a\|\gamma_*\|$, we obtain from (4.8)

$$\left\| \begin{bmatrix} d \\ \gamma - \gamma_* \end{bmatrix} \right\| \leq \kappa_o \delta_1 \epsilon. \tag{4.13}$$

Let us now write γ in terms of components whose dimensions are compatible with those of γ_* in (4.6), i.e.,

$$\gamma = \begin{bmatrix} \lambda \\ \nu \end{bmatrix}. \tag{4.14}$$

Then, from (4.13)

$$\|d\| \leq \kappa_o \delta_1 \epsilon \quad \text{and} \quad \|\nu - [\mu_*]_{\mathcal{W}_*}\| \leq \kappa_o \delta_1 \epsilon.$$

Hence,

$$\begin{aligned} \nu &= [\mu_*]_{\mathcal{W}_*} - ([\mu_*]_{\mathcal{W}_*} - \nu) \\ &\geq [\mu_*]_{\mathcal{W}_*} - \kappa_o \delta_1 \epsilon \mathbf{1}, \end{aligned} \tag{4.15}$$

and

$$\begin{aligned} g_{\mathcal{W}_*^c}(x_k) + G_{\mathcal{W}_*^c}(x_k)d &= g_{\mathcal{W}_*^c}(x_*) - (g_{\mathcal{W}_*^c}(x_*) - g_{\mathcal{W}_*^c}(x_k)) \\ &\quad - (G_{\mathcal{W}_*^c}(x_*) - G_{\mathcal{W}_*^c}(x_k))d + G_{\mathcal{W}_*^c}(x_*)d \\ &\geq g_{\mathcal{W}_*^c}(x_*) - \big(\kappa_g + \kappa_G \kappa_o \delta_1 \epsilon + \|G_{\mathcal{W}_*^c}(x_*)\|\kappa_o \delta_1\big)\, \epsilon \mathbf{1}, \end{aligned} \tag{4.16}$$

where κ_g and κ_G are, respectively, Lipschitz constants for $g_{\mathcal{W}_*^c}(x_*)$ and $G_{\mathcal{W}_*^c}(x)$ over $\mathcal{N}_*(\epsilon)$ and $\mathbf{1}$ is a vector of all ones of appropriate dimension. Therefore, if ϵ is small enough we have from (4.11), (4.15), (4.16) that

$$\nu > 0, \quad g_{\mathcal{W}_*^c}(x_k) + G_{\mathcal{W}_*^c}(x_k)d > 0. \tag{4.17}$$

We can now construct the solution of the IQP subproblem as follows

$$d_k^{\mathrm{IQ}} = d, \quad \begin{bmatrix} \lambda_{k+1}^{\mathrm{IQ}} \\ [\mu_{k+1}^{\mathrm{IQ}}]_{\mathcal{W}_*} \end{bmatrix} = \gamma = \begin{bmatrix} \lambda \\ \nu \end{bmatrix}, \quad [\mu_{k+1}^{\mathrm{IQ}}]_{\mathcal{W}_*^c} = 0. \tag{4.18}$$

By (4.4) and (4.17), it is easy to verify that this primal-dual solution satisfies the KKT conditions (3.2) of the IQP subproblem. Therefore, the vector d_k^{IQ} constructed in this manner is indeed the unique primal optimal solution of the IQP subproblem and its working set satisfies $\mathcal{W}_k = \mathcal{W}_*$. □

We now present the main result of this section. It shows that, if the algorithm takes the full step for all (x_k, λ_k, μ_k) is sufficiently close to (x_*, λ_*, μ_*), the iteration is quadratically convergent.

Theorem 4.3 *Suppose that: i) Assumptions 4.1 hold; ii) For (x_k, λ_k, μ_k) sufficiently close to (x_*, λ_*, μ_*), Algorithm I computes the step d_k by (2.13) with $\alpha_k = 1$. Then, there exists a neighborhood $\mathcal{N}_{**}(\epsilon)$ such that if $(x_k, \lambda_k, \mu_k) \in \mathcal{N}_{**}(\epsilon)$, the sequence $\{(x_l, \lambda_l, \mu_l)\}_{l=k}^{\infty}$ converges quadratically to (x_*, λ_*, μ_*).*

Proof. By Lemma 4.2, if x_k is sufficiently close to x_* we have $\mathcal{W}_k = \mathcal{W}_*$. The KKT conditions of the EQP subproblem (2.3) are therefore given by

$$\begin{bmatrix} W_k & -A_{\mathcal{W}_*}^T(x_k) \\ A_{\mathcal{W}_*}(x_k) & 0 \end{bmatrix} \begin{bmatrix} d \\ \theta \end{bmatrix} = - \begin{bmatrix} \nabla f_k + W_k d_k^{\mathrm{IQ}} \\ 0 \end{bmatrix}. \tag{4.19}$$

By Assumptions 4.1-(b),(e), if x_k is sufficiently near x_*, the linear system (4.19) is nonsingular and its solution is thus given by

$$(d, \theta) = (d_k^{\mathrm{EQ}}, \theta_{k+1}) \quad \text{where} \quad \theta_{k+1} \triangleq \begin{bmatrix} \lambda_{k+1}^{\mathrm{EQ}} \\ \mu_{k+1}^{\mathrm{EQ}} \end{bmatrix}. \tag{4.20}$$

Let us rewrite (4.19) as

$$\begin{bmatrix} W_k & -A_{\mathcal{W}_*}^T(x_k) \\ A_{\mathcal{W}_*}(x_k) & 0 \end{bmatrix} \begin{bmatrix} d + d_k^{\mathrm{IQ}} \\ \theta \end{bmatrix} = - \begin{bmatrix} \nabla f_k \\ c_{\mathcal{W}_*}(x_k) \end{bmatrix}, \tag{4.21}$$

and note that the inverse of the coefficient matrix is bounded above in norm by some constant δ_2, for all x_k close to x_*. Now, since (4.21) has the same form as (4.4), except that B_k is replaced by W_k, we can follow the same reasoning as in the proof of Lemma 4.2 and deduce that the solution $(d_k^{\mathrm{EQ}} + d_k^{\mathrm{IQ}}, \theta_{k+1})$ of (4.21) also satisfies (4.17), i.e.,

$$\mu_{k+1}^{\mathrm{EQ}} > 0 \quad \text{and} \quad g_{\mathcal{W}_*^c}(x_k) + G_{\mathcal{W}_*^c}(x_k)(d_k^{\mathrm{IQ}} + d_k^{\mathrm{EQ}}) > 0. \tag{4.22}$$

Therefore, for (x_k, λ_k, μ_k) sufficiently close to (x_*, λ_*, μ_*), step 4 of Algorithm I will choose $\beta = 1$, step 6 will set

$$d_k = d_k^{\mathrm{IQ}} + d_k^{\mathrm{EQ}} \quad \text{and} \quad x_{k+1} - x_k = d_k^{\mathrm{IQ}} + d_k^{\mathrm{EQ}}, \tag{4.23}$$

and step 7 will set

$$\lambda_{k+1} = \lambda_{k+1}^{\mathrm{EQ}}, \quad [\mu_{k+1}]_{\mathcal{W}_*} = \mu_{k+1}^{\mathrm{EQ}}, \quad [\mu_{k+1}]_{\mathcal{W}_*^c} = 0. \tag{4.24}$$

Let us define

$$\hat{W}(x,\theta) = \nabla^2 f(x) - \sum_{i\in\mathcal{E}} \lambda_i^{\mathrm{EQ}} \nabla^2 h_i(x) - \sum_{i\in\mathcal{I}\cap\mathcal{W}_*} \mu_i^{\mathrm{EQ}} \nabla^2 g_i(x) \quad \text{with} \quad \theta = \begin{bmatrix} \lambda^{\mathrm{EQ}} \\ \mu^{\mathrm{EQ}} \end{bmatrix}. \tag{4.25}$$

For (x_k, λ_k, μ_k) sufficiently close to (x_*, λ_*, μ_*), we have by (4.24) that $\hat{W}(x_k, \theta_k) = W_k$ (see (2.4)), and thus recalling (4.23) we can write (4.21) as

$$\begin{bmatrix} \hat{W}(x_k,\theta_k) & -A_{\mathcal{W}_*}^T(x_k) \\ A_{\mathcal{W}_*}(x_k) & 0 \end{bmatrix} \begin{bmatrix} x_{k+1} - x_k \\ \theta_{k+1} - \theta_k \end{bmatrix} = - \begin{bmatrix} \nabla f(x_k) - A_{\mathcal{W}_*}^T(x_k)\theta_k \\ c_{\mathcal{W}_*}(x_k) \end{bmatrix}. \tag{4.26}$$

Note that this is the Newton iteration applied to the nonlinear system

$$\nabla f(x) - A_{\mathcal{W}_*}^T(x)\theta = 0 \tag{4.27a}$$

$$c_{\mathcal{W}_*}(x) = 0. \tag{4.27b}$$

We can now perform standard Newton analysis to establish quadratic convergence. We first observe that the matrix

$$F(x,\theta) \triangleq \begin{bmatrix} \hat{W}(x,\theta) & -A_{\mathcal{W}_*}^T(x) \\ A_{\mathcal{W}_*}(x) & 0 \end{bmatrix}$$

is continuous by Assumptions 4.1-(a). Thus F is also Lipschitz continuous near x_*, with Lipschitz constant κ_1. If we define

$$\theta_* = \begin{bmatrix} \lambda_* \\ [\mu_*]_{\mathcal{W}_*} \end{bmatrix}, \tag{4.28}$$

then by assumptions 4.1-(e), F is nonsingular in a neighborhood of (x_*, θ_*). Hence F is invertible and F^{-1} is bounded above in norm by a constant κ_2 in a neighborhood of (x_*, θ_*). By Theorem 5.2.1 of [14], if (x_k, θ_k) satisfies

$$\left\| \begin{bmatrix} x_k - x_* \\ \theta_k - \theta_* \end{bmatrix} \right\| \leq \frac{1}{2\kappa_1\kappa_2}, \tag{4.29}$$

then

$$\left\| \begin{bmatrix} x_{k+1} - x_* \\ \theta_{k+1} - \theta_* \end{bmatrix} \right\| \leq \kappa_1\kappa_2 \left\| \begin{bmatrix} x_k - x_* \\ \theta_k - \theta_* \end{bmatrix} \right\|^2. \tag{4.30}$$

Let us define the neighborhood

$$\mathcal{N}_{**}(\epsilon) = \left\{ (x,\lambda,\mu) \;\middle|\; \left\| \begin{bmatrix} x - x_* \\ \lambda - \lambda_* \\ \mu - \mu_* \end{bmatrix} \right\| \leq \min\left(\epsilon, \frac{1}{2\kappa_1\kappa_2}\right) \right\},$$

where $\mu \in \mathbb{R}^t$ and where $\epsilon > 0$ is small enough that all the properties derived so far in this proof hold for $(x_k, \lambda_k, \mu_k) \in \mathcal{N}_{**}(\epsilon)$. In particular, by (4.24), we have that $[\mu_{k+1}]_{\mathcal{W}_*^c} = [\mu_*]_{\mathcal{W}_*^c} = 0$. Thus, recalling the definitions (4.20), (4.28), and using (4.30), we have that if $(x_k, \lambda_k, \mu_k) \in \mathcal{N}_{**}(\epsilon)$,

$$\begin{aligned}
\left\| \begin{bmatrix} x_{k+1} - x_* \\ \lambda_{k+1} - \lambda_* \\ \mu_{k+1} - \mu_* \end{bmatrix} \right\| &= \left\| \begin{bmatrix} x_{k+1} - x_* \\ \theta_{k+1} - \theta_* \end{bmatrix} \right\| \\
&\leq \kappa_1 \kappa_2 \left\| \begin{bmatrix} x_k - x_* \\ \theta_k - \theta_* \end{bmatrix} \right\|^2 \\
&\leq \kappa_1 \kappa_2 \left\| \begin{bmatrix} x_k - x_* \\ \lambda_k - \lambda_* \\ \mu_k - \mu_* \end{bmatrix} \right\|^2 \\
&\leq \min \left(\kappa_1 \kappa_2 \epsilon^2, \frac{1}{4 \kappa_1 \kappa_2} \right).
\end{aligned} \tag{4.31}$$

If ϵ is sufficiently small such that $\kappa_1 \kappa_2 \epsilon^2 \leq \epsilon$, we have that $(x_{k+1}, \lambda_{k+1}, \mu_{k+1}) \in \mathcal{N}_{**}(\epsilon)$. Therefore, we have shown that if $(x_k, \lambda_k, \mu_k) \in \mathcal{N}_{**}(\epsilon)$, all subsequent iterates remain in $\mathcal{N}_{**}(\epsilon)$ and converge quadratically to (x_*, λ_*, μ_*). □

This quadratic convergence result is more satisfying than those established in the literature of SQP methods that use exact Hessians. This is because, even in a neighborhood of a solution satisfying Assumptions 4.1, the quadratic program (2.1) may have several local minimizers if B_k is replaced by the Hessian W_k. Thus, for classical SQP methods it is necessary to assume that the quadratic programming solver selects a local with certain properties; for example, the one closest to the current iterate. Our quadratic convergence result, relies only on the second-order sufficiency conditions of the problem.

To establish Theorem 4.3, we have assumed that near the solution the line search condition (2.12) is satisfied for $\alpha_k = 1$. As is well known, however, the nonsmooth penalty function ϕ_π may reject unit steplengths (the Maratos effect) and some additional features must be incorporated into the algorithm to prevent this from happening. They include second-order corrections or watchdog steps; see e.g. [30].

5 Implementation and Numerical Results

The SQP+ algorithm can be implemented as an extension of any SQP method that solves convex quadratic programs for the step computation. Our implementation is based on SQPLAB, a Matlab package developed by Gilbert [21] that offers a classical line search SQP method using an ℓ_1 merit function and BFGS quasi-Newton updating. SQPLAB does not contain many of the algorithmic features of production SQP codes such as SNOPT [23, 22] or FILTERSQP [19], and is not well suited for large scale problems. Nevertheless, it provides a good platform for the development of a prototype implementation of the SQP+ approach that allows us to study the benefits of the EQP phase on small and medium size problems.

We made two modifications to SQPLAB in order to improve its performance. Instead of using The Mathworks' routine `quadprog` to solve the quadratic program (2.1), we employed KNITRO/ACTIVE [8] for this task. Unlike `quadprog`, the KNITRO/ACTIVE code had no difficulties solving the quadratic subproblems generated during the IQP phase, and provided reliable multiplier estimates. The second modification concerns the choice of the penalty parameter π, which in SQPLAB is based on the size of Lagrange multiplier estimates. We replaced this technique by the rule (2.10)-(2.11) and observed a notable improvement in performance.

The BFGS quasi-Newton approximation B_k used in the IQP phase of the method is updated using information from the full step. We define the correction pairs as

$$s_k = x_{k+1} - x_k \quad \text{and} \quad y_k = \nabla_x L(x_{k+1}, \lambda_{k+1}, \mu_{k+1}) - \nabla_x L(x_k, \lambda_{k+1}, \mu_{k+1}),$$

and modify y_k, if necessary, by means of Powell's damping procedure [33] to ensure that all Hessian approximations B_k are positive definite.

The gradients of the constraints in the working set $\mathcal{W}_k$ (defined in (2.2)) may not be linearly independent, and this can cause difficulties in the EQP step computation. In a production implementation of the SQP+ algorithm, we would define $\mathcal{W}_k$ to be a subset of (2.2) that has (sufficiently) linearly independent constraint gradients. In our simple implementation of SQP+, we choose to simply skip the EQP phase if the constraints gradients in $\mathcal{W}_k$ are dependent (or nearly so).

As mentioned in Section 2, we can compute an approximate solution of the EQP problem (2.3) by either applying the projected conjugate gradient method [30] (and adding a trust region constraint to (2.3)) or using a direct method. Here we follow the latter approach. When the EQP problem (2.3) is convex, its solution solves the KKT system

$$\begin{bmatrix} W_k & H^T(x_k) & G_{\mathcal{W}_k}^T(x_k) \\ H(x_k) & 0 & 0 \\ G_{\mathcal{W}_k}(x_k) & 0 & 0 \end{bmatrix} \begin{bmatrix} d^{\mathrm{EQ}} \\ \lambda^{\mathrm{EQ}} \\ \mu^{\mathrm{EQ}} \end{bmatrix} = - \begin{bmatrix} \nabla f(x_k) + W_k d^{\mathrm{IQ}} \\ 0 \\ 0 \end{bmatrix}, \tag{5.1}$$

where, as in the previous section, $G_{\mathcal{W}_k}$ is the matrix obtained by selecting the rows of G corresponding to the elements of $\mathcal{W}_k$. If the inertia of the KKT matrix in (5.1) is given by

$$(n, |\mathcal{W}_k|, 0), \tag{5.2}$$

then we know that problem (2.3) is convex [30]. If this is not the case, we could repeatedly add an increasingly large multiple of the identity matrix to W_k, as discussed in [37], until the inertia of the modified system is given by (5.2). We have tested such an approach, but found this modification procedure to be a heuristic that is not always satisfactory. Therefore in keeping with our minimalist implementation of the SQP+ method, we simply skip the EQP step if the inertia of (5.1) is not given by (5.2).

In the line search procedure, we only test the unit steplength along the full step $d^{\mathrm{IQ}} + \beta d^{\mathrm{EQ}}$. If $\alpha_k = 1$ does not yield the sufficient decrease condition (2.12), we simply fall back on the IQP step and commence the backtracking line search from there. The algorithm

terminates when the following conditions are satisfied

$$\begin{aligned} \|\nabla f(x_k) - H(x_k)^T\lambda_k - G(x_k)^T\mu_k\|_2 &\leq \epsilon_1 \\ \|\big(h(x_k), \max[0, -g(x_k)]\big)\|_2 &\leq \epsilon_2 \\ \|\mu_k^T g(x_k)\|_2 &\leq \epsilon_3, \end{aligned} \tag{5.3}$$

for some constants $\epsilon_1, \epsilon_2, \epsilon_3$. We can now give a detailed description of the algorithm used in our tests.

Algorithm SQP+

Initial data: $x_0, \pi_0 > 0$, $\pi_b > 0$, $\epsilon_1 > 0$, $\epsilon_2 > 0$, $\epsilon_3 > 0$, $\sigma > 0$, $\tau \in (0,1)$ and $B_1 = I$

For $k = 1, 2, ...$ until (x_k, λ_k, μ_k) satisfies the termination test (5.3):

1. Compute the step $d^{\rm IQ}$ and multipliers $\lambda^{\rm IQ}$, $\mu^{\rm IQ}$ by solving (2.1).
2. Determine the working set $\mathcal{W}_k$. If $|\mathcal{W}_k| \geq n$, go to step 7.
3. Factor the system (5.1); if its inertia is not given by (5.2), then go to step 7.
   Else, compute the EQP step $d^{\rm EQ}$ and multipliers $\lambda^{\rm EQ}$, $\mu^{\rm EQ}$ by solving the linear system (5.1).
4. Compute $\beta \geq 0$ to be the largest number in $[0,1]$ such that $d = d^{\rm IQ} + \beta d^{\rm EQ}$ satisfies (2.1*b*), (2.1*c*).
5. Update the penalty parameter π_k by the rule (2.10)-(2.11).
6. If the sufficient decrease condition (2.12) is satisfied, for $\alpha_k = 1$, then set
$$x_{k+1} = x_k + d^{\rm IQ} + \beta d^{\rm EQ},$$
$$\lambda_{k+1} = \lambda_{k+1}^{\rm EQ}, \quad [\mu_{k+1}]_{\mathcal{W}_k} = \max\left(0, \mu_{k+1}^{\rm EQ}\right), \quad [\mu_{k+1}]_{\mathcal{W}_k^c} = 0,$$
   and go to step 8.
7. Let $\alpha_k \in (0,1]$ be the first member of the sequence $\{1, \tau, \tau^2, \ldots\}$ such that
$$\phi_{\pi_k}(x_k + \alpha_k d^{\rm IQ}) \leq \phi_{\pi_k}(x_k) - \sigma\alpha_k qred(d^{\rm IQ}), \tag{5.4}$$
   and set
$$x_{k+1} = x_k + \alpha_k d^{\rm IQ}; \quad (\lambda_{k+1}, \mu_{k+1}) = \big((1-\alpha_k)\lambda_k + \alpha_k\lambda^{\rm IQ}, (1-\alpha_k)\mu_k + \alpha_k\mu^{\rm IQ}\big). \tag{5.5}$$
8. Update B_{k+1} from B_k using the BFGS method with Powell damping.

The constants in the algorithm are chosen as follows: $\epsilon_1 = \epsilon_2 = \epsilon_3 = 10^{-5}$, $\sigma = 10^{-4}$, $\tau = 0.5$. Initially, $\pi_b = 0$ and at the end of the first iteration it is reset to $\pi_b = \max(\sqrt{\epsilon_m}, \|(\lambda_1, \mu_1)\|_\infty/100)$.

5.1 Numerical Experiments

We tested Algorithm SQP+ on a set of problems from the CUTEr [26] collection. Our test set consists primarily of small-dimensional problems, but we have included also a significant number of medium-size problems. The characteristics of the problems are given in Tables 1-4 of the Appendix. The test set was selected before the numerical experiments were performed; no problems were removed or added during the testing process.

We compared the performance of SQP+ and SQPLAB (with the improvements described above) in terms of the number of iterations needed to achieve convergence. The results are reported in Figure 5.1 using the logarithmic performance profiles described in [15]. (A comparison based on number of function evaluations would give very similar performance profiles.) Detailed results are given in Tables 1-4.

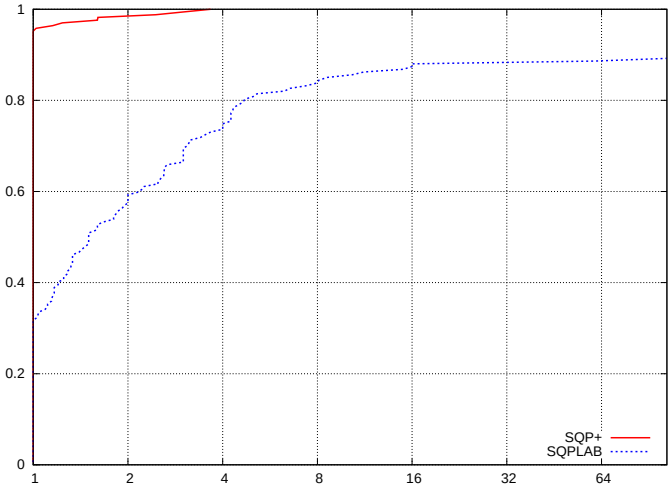

Figure 1: Comparison of SQPLAB and SQP+ in terms of iterations.

The benefit of adding the EQP phase can be observed by comparing SQP+ and SQPLAB since the other aspects of these two algorithms are identical. The results presented here strongly suggest that this benefit can be quite substantial. Our Matlab implementation does not permit timing experiments on large scale problems, therefore an evaluation of the speed of the SQP+ approach must await the development of a sparse large-scale implementation. We mention, however, that the additional computational cost incurred by the EQP phase is not likely to be significant. This view is based on the numerical experiments reported in [4] using a sequential linear-quadratic programming method that contains an EQP phase, just like Algorithm SQP+. That paper reports that the cost of the EQP phase is dominated by the cost of the linear programming phase. We can expect the same in the SQP+ algorithm since solving a quadratic program is generally more expensive than solving a linear program.

To obtain another measure of the performance of the SQP+ algorithm, we also compare its performance with SNOPT version 7.2-1 in terms of major iterations. We used the same test set as in the previous experiment except that we removed all linear and quadratic

programs because SNOPT recognizes their structure and solves them in one major iteration (e.g. using the exact Hessian for a quadratic program) while the SQP+ method treats linear and quadratic programs as general nonlinear programs. The results are reported in Figure 5.1. One should be cautious in interpreting these results because they are obtained using substantially different implementations of the SQP approach, which employ different scalings and termination tests. SNOPT is a mature code with many features and enhancements not present in SQP+. The results are, nevertheless, highly encouraging both in terms of robustness and number of iterations.

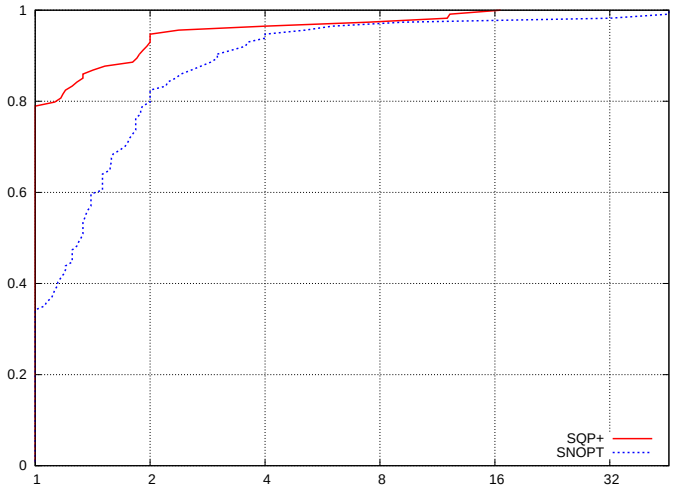

Figure 2: Comparison of the number of major iterations in SNOPT and SQP+.

We conclude this section by describing an enhancement to the EQP phase that we developed and tested. Rather than using the contraction parameter β in step 4 of Algorithm SQP+ to ensure that all linearized constraints are satisfied, we experimented with the option of computing the minimum norm projection of the step $d^{\mathrm{IQ}} + d^{\mathrm{EQ}}$ onto the feasible region defined by (2.1*b*), (2.1*c*). We found that the improvements in performance obtained with the projection approach were not substantial, except on bound constrained problems. Since computing the projection is expensive, this option does not seem viable at this point and further investigation is needed to determine what is the most effective implementation of the EQP phase.

6 Final Remarks

We have presented a sequential quadratic programming method that can employ exact second derivative information but never solves indefinite quadratic programs. It is a two-stage method in which global convergence and active-set identification are driven by an IQP phase (which solves convex quadratic programs), while fast asymptotic convergence is achieved by an EQP phase (which solves equality constrained subproblems). The numerical

results presented in this paper suggest that the addition of the EQP phase leads to an important decrease in the total number of iterations and function evaluations.

There is considerable flexibility in the SQP+ approach. For example, if a projected CG iteration is employed in the EQP phase, the Hessian of the Lagrangian need not be formed; only products of this Hessian times vectors are required. Moreover, one could use gradient differences to approximate these products, and in this case the SQP+ method would only require first derivatives.

The approach presented here does not overcome one of the main limitations of SQP methods, namely the major computational cost of solving large quadratic programs, particularly when the reduced space is large. Although the IQP matrix B_k can be chosen to be a simple matrix such as a limited memory BFGS matrix, or even a diagonal matrix, the cost of the IQP phase can still be significant. Therefore we view SQP+ as a method that is applicable to the class of problems that is currently solved by SQP methods. An interesting alternative approach for improving SQP methods is reported in [28].

Acknowledgement. We would like to thank Richard Byrd for many useful comments during the course of this research.

References

[1] P. T. Boggs and J. W. Tolle. Sequential quadratic programming. *Acta Numerica*, 4:1–51, 1995.

[2] J. F. Bonnans and G. Launay. Sequential quadratic-programming with penalization of the displacement. *SIAM Journal on Optimization*, 5(4):792–812, 1995.

[3] J. V. Burke and S. P. Han. A robust sequential quadratic-programming method. *Mathematical Programming*, 43(3):277–303, 1989.

[4] R. H. Byrd, N. I. M. Gould, J. Nocedal, and R. A. Waltz. An algorithm for nonlinear optimization using linear programming and equality constrained subproblems. *Mathematical Programming, Series B*, 100(1):27–48, 2004.

[5] R. H. Byrd, N. I. M. Gould, J. Nocedal, and R. A. Waltz. On the convergence of successive linear-quadratic programming algorithms. *SIAM Journal on Optimization*, 16(2):471–489, 2006.

[6] R. H. Byrd, G. López-Calva, and J. Nocedal. A line search penalty method for nonlinear optimization. Technical Report 08/05, Optimization Technology Center, Northwestern University, 2008.

[7] R. H. Byrd, J. Nocedal, and R. A. Waltz. Steering exact penalty methods. *Optimization Methods and Software*, 23(2), 2008.

[8] R. H. Byrd, J. Nocedal, and R.A. Waltz. KNITRO: An integrated package for nonlinear optimization. In G. di Pillo and M. Roma, editors, *Large-Scale Nonlinear Optimization*, pages 35–59. Springer, 2006.

[9] R. M. Chamberlain, M. J. D. Powell, C. Lemaréchal, and H. C. Pedersen. The watchdog technique for forcing convergence in algorithms for constrained optimization. *Mathematical Programming Studies*, 16(MAR):1–17, 1982.

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

[11] C. M. Chin and R. Fletcher. On the global convergence of an SLP-filter algorithm that takes EQP steps. *Mathematical Programming, Series A*, 96(1):161–177, 2003.

[12] T. F. Coleman and A. Verma. A preconditioned conjugate gradient approach to linear equality constrained minimization. Technical report, Computer Science Department and Cornell Theory Center, Cornell University, Ithaca, NY 14850, USA, July 1998.

[13] A. R. Conn, N. I. M. Gould, and Ph. Toint. *Trust-region methods*. MPS-SIAM Series on Optimization. SIAM publications, Philadelphia, Pennsylvania, USA, 2000.

[14] J. E. Dennis and R. B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice-Hall, Englewood Cliffs, New Jersey, USA, 1983. Reprinted as *Classics in Applied Mathematics 16*, SIAM, Philadelphia, Pennsylvania, USA, 1996.

[15] E. D. Dolan and J. J. Moré. Benchmarking optimization software with performance profiles. *Mathematical Programming, Series A*, 91:201–213, 2002.

[16] A. Drud. CONOPT: A GRG code for large sparse dynamic nonlinear optimization problems. *Mathematical Programming*, 31:153–191, 1985.

[17] F. Facchinei and S. Lucidi. Quadratically and superlinearly convergent algorithms for the solution of inequality constrained minimization problems. *Journal of Optimization Theory and Applications*, 85:265–289, 1994.

[18] R. Fletcher. *Practical Methods of Optimization*. J. Wiley and Sons, Chichester, England, second edition, 1987.

[19] R. Fletcher and S. Leyffer. Nonlinear programming without a penalty function. *Mathematical Programming*, 91:239–269, 2002.

[20] R. Fletcher and E. Sainz de la Maza. Nonlinear programming and nonsmooth optimization by successive linear programming. *Mathematical Programming*, 43(3):235–256, 1989.

[21] J. C. Gilbert. SQPlab - A MATLAB software package for solving nonlinear optimization problems and optimal control problems. Technical Report Version 0.4.1, INRIA, 2007.

[22] P. E. Gill, W. Murray, and M. A. Saunders. SNOPT: An SQP algorithm for large-scale constrained optimization. *SIAM Journal on Optimization*, 12:979–1006, 2002.

[23] P. E. Gill, W. Murray, and M. A. Saunders. User's guide for SNOPT 7.1: a Fortran package for large-scale nonlinear programming. Technical Report NA 05-2, Department of Mathematics, University of California, San Diego, 2005.

[24] P. E. Gill, W. Murray, and M. H. Wright. *Practical Optimization*. Academic Press, London, 1981.

[25] N. I. M. Gould, M. E. Hribar, and J. Nocedal. On the solution of equality constrained quadratic problems arising in optimization. *SIAM Journal on Scientific Computing*, 23(4):1375–1394, 2001.

[26] N. I. M. Gould, D. Orban, and Ph. L. Toint. CUTEr (and SifDec), a Constrained and Unconstrained Testing Environment, revisited. Technical Report TR/PA/01/04, CERFACS, Toulouse, France, 2003. To appear in Transactions on Mathematical Software.

[27] N. I. M. Gould, D. Orban, and Ph. L. Toint. Numerical methods for large-scale nonlinear optimization. *Acta Numerica*, pages 299–361, 2005.

[28] N. I. M. Gould and D. P. Robinson. A second derivative SQP method with imposed descent. Technical Report 08/09, Oxford University Computing Laboratory, 2008.

[29] C. Keller, N. I. M. Gould, and A. J. Wathen. Constraint preconditioning for indefinite linear systems. *SIAM Journal on Matrix Analysis and Applications*, 21(4):1300–1317, 2000.

[30] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer Series in Operations Research. Springer, 1999.

[31] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer Series in Operations Research. Springer, second edition, 2006.

[32] E. O. Omojokun. *Trust region algorithms for optimization with nonlinear equality and inequality constraints*. PhD thesis, University of Colorado, Boulder, Colorado, USA, 1989.

[33] M. J. D. Powell. A fast algorithm for nonlinearly constrained optimization calculations. In G. A. Watson, editor, *Numerical Analysis, Dundee 1977*, number 630 in Lecture Notes in Mathematics, pages 144–157, Heidelberg, Berlin, New York, 1978. Springer Verlag.

[34] M. J. D. Powell. Variable metric methods for constrained optimization. In Bachem, A., Grötschel, M., and Korte, B., editors, *Mathematical Programming : The State of the Art, Bonn 1982*. Springer-Verlag, 1983.

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

[36] K. Schittkowski. The nonlinear programming method of Wilson, Han and Powell with an augmented Lagrangian type line search function. *Numerische Mathematik*, 38:83–114, 1981.

[37] R. J. Vanderbei and D. F. Shanno. An interior point algorithm for nonconvex nonlinear programming. *Computational Optimization and Applications*, 13:231–252, 1999.

[38] R. A. Waltz, J. L. Morales, J. Nocedal, and D. Orban. An interior algorithm for nonlinear optimization that combines line search and trust region steps. *Mathematical Programming, Series A*, 107:391–408, 2006.

Appendix: Complete Numerical Results

name	n	bounds	ineq	equal	SQP+	SQPLAB
airport	84	168	42	0	11	89
biggsb1	144	286	0	0	72	188
bqpgasim	50	100	0	0	3	24
chenhark	200	200	0	0	4	413
clnlbeam	200	198	0	0	3	3
combustion	144	288	0	0	1	15
dixchlnv	100	100	0	50	28	-24
dnieper	57	112	0	24	23	-49
dual3	111	222	0	1	13	56
eg3	101	200	199	1	18	18
eigmaxa	101	200	0	101	6	6
eigmaxb	101	202	0	101	8	8
eigmina	101	202	0	101	2	2
eigminb	101	202	0	101	8	8
expfita	5	0	21	0	13	39
explin	144	288	0	0	97	112
explin2	144	288	0	0	72	95
grouping	100	200	0	125	1	1
haifas	7	0	9	0	10	10
haldmads	6	0	42	0	9	10
hanging	288	0	180	0	46	162
harkerp2	144	144	0	0	37	92
himmelbi	100	100	12	0	35	36
hs001	2	1	0	0	26	21
hs002	2	1	0	0	10	15
hs003	2	1	0	0	2	8
hs004	2	2	0	0	2	2
hs005	2	4	0	0	8	5
hs006	2	0	0	1	7	7
hs007	2	0	0	1	9	10
hs008	2	0	0	2	5	5
hs009	2	0	0	1	5	5
hs010	2	0	1	0	9	12
hs011	2	0	1	0	6	11
hs012	2	0	1	0	8	10
hs014	2	0	1	1	6	7
hs015	2	1	2	0	3	-7
hs018	2	4	2	0	6	8

Table 1: Number of iterations for SQP+ and SQPLAB. A negative number indicates failure to converge.

name	n	bounds	ineq	equal	SQP+	SQPLAB
hs019	2	4	2	0	6	6
hs020	2	2	3	0	8	-22
hs021	2	4	1	0	1	3
hs022	2	0	2	0	4	4
hs023	2	4	5	0	6	6
hs024	2	2	2	0	4	4
hs025	3	6	0	0	0	0
hs026	3	0	0	1	21	21
hs027	3	0	0	1	24	32
hs028	3	0	0	1	4	4
hs029	3	0	1	0	7	10
hs030	3	6	1	0	1	2
hs031	3	6	1	0	6	-10
hs032	3	3	1	1	3	4
hs033	3	4	2	0	4	4
hs034	3	6	2	0	7	7
hs035	3	3	1	0	3	9
hs036	3	6	1	0	2	-3
hs037	3	6	1	0	5	8
hs038	4	8	0	0	60	100
hs039	4	0	0	2	12	15
hs040	4	0	0	3	6	6
hs041	4	8	0	1	5	6
hs042	3	3	0	1	5	9
hs043	4	0	3	0	7	9
hs044	4	4	6	0	6	6
hs045	5	10	0	0	0	0
hs046	5	0	0	2	51	51
hs047	5	0	0	3	28	29
hs048	5	0	0	2	7	7
hs049	5	0	0	2	29	29
hs050	5	0	0	3	16	16
hs051	5	0	0	3	3	3
hs052	5	0	0	3	8	12
hs053	5	10	0	3	1	16
hs054	6	12	0	1	2	5
hs055	6	8	0	6	2	2
hs056	7	7	0	4	5	13
hs057	2	2	1	0	6	2
hs059	2	4	3	0	10	15
hs060	3	6	0	1	6	9
hs062	3	6	0	1	7	9
hs064	3	3	1	0	35	110
hs065	3	6	1	0	8	25
hs066	3	6	2	0	4	6

Table 2: Number of iterations for SQP+ and SQPLAB. A negative number indicates failure to converge.

name	n	bounds	ineq	equal	SQP+	SQPLAB
hs070	4	8	1	0	14	31
hs071	4	8	1	1	5	6
hs072	4	8	2	0	18	35
hs073	4	4	2	1	4	4
hs075	4	8	1	3	6	18
hs076	4	4	3	0	1	5
hs077	5	0	0	2	17	17
hs078	5	0	0	3	7	7
hs079	5	0	0	3	13	13
hs080	5	10	0	3	6	7
hs081	5	10	0	3	15	13
hs083	5	10	3	0	4	-6
hs085	5	10	36	0	0	0
hs086	5	5	6	0	5	9
hs087	9	18	0	4	67	88
hs089	3	0	1	0	23	60
hs090	4	0	1	0	24	25
hs091	5	0	1	0	18	39
hs092	6	0	1	0	26	38
hs093	6	6	2	0	76	31
hs095	6	12	4	0	2	2
hs096	6	12	4	0	2	2
hs097	6	12	4	0	13	13
hs098	6	12	4	0	13	13
hs100	7	0	4	0	8	16
hs100lnp	7	0	0	2	14	16
hs101	7	14	6	0	49	-77
hs102	7	14	6	0	32	-91
hs103	7	14	6	0	39	245
hs104	8	16	6	0	11	34
hs105	8	16	0	0	75	-94
hs106	8	16	6	0	24	46
hs108	9	1	13	0	11	12
hs110	10	20	0	0	6	7
hs111	10	20	0	3	10	47
hs111lnp	10	0	0	3	48	47
hs113	10	0	8	0	5	17
hs116	13	26	15	0	348	95
hs117	15	15	5	0	17	-17
hs118	15	30	17	0	7	13
hs119	16	32	0	8	4	-19
hs268	5	0	5	0	2	32
hs3mod	2	1	0	0	5	7
hs44new	4	4	5	0	5	5

Table 3: Number of iterations for SQP+ and SQPLAB. A negative number indicates failure to converge.

name	n	bounds	ineq	equal	SQP+	SQPLAB
jnlbrng1	144	144	0	0	7	36
jnlbrng2	144	144	0	0	4	45
jnlbrnga	144	144	0	0	3	31
jnlbrngb	144	144	0	0	1	61
loadbal	31	42	20	11	17	79
makela3	21	0	20	0	22	-19
mccormck	144	288	0	0	4	17
minsurfo	144	144	0	0	5	43
ncvxbqp1	144	288	0	0	2	-3
ncvxbqp2	144	288	0	0	4	-6
ncvxbqp3	144	288	0	0	124	142
nobndtor	144	144	0	0	5	37
nonscomp	144	288	0	0	9	59
obstclae	169	338	0	0	11	28
obstclbm	169	338	0	0	6	26
optctrl3	118	0	1	79	7	-206
optctrl6	118	0	1	79	7	-206
optprloc	30	60	29	0	6	12
pentdi	144	144	0	0	1	2
probpenl	144	288	0	0	2	2
prodpl0	60	60	9	20	8	8
prodpl1	60	60	9	20	6	6
qrtquad	144	20	0	0	36	57
qudlin	144	288	0	0	2	-3
rk23	17	6	0	11	9	9
s368	144	288	0	0	0	0
synthes1	6	12	6	0	5	8
torsion1	144	288	0	0	4	17
torsion2	144	288	0	0	5	20
torsion3	144	288	0	0	2	9
torsion4	144	288	0	0	5	13
torsion5	144	288	0	0	1	3
torsion6	144	288	0	0	3	11
torsiona	144	288	0	0	4	17
torsionb	144	288	0	0	8	18
torsionc	144	288	0	0	2	8
torsiond	144	288	0	0	4	17
torsione	144	288	0	0	1	3
torsionf	144	288	0	0	3	8
trimloss	142	264	52	20	183	114

Table 4: Number of iterations for SQP+ and SQPLAB. A negative number indicates failure to converge.