ON THE PERFORMANCE OF SQP METHODS FOR NONLINEAR OPTIMIZATION

PHILIP E. GILL*, MICHAEL A. SAUNDERS†, AND ELIZABETH WONG*

Abstract. This paper concerns some practical issues associated with the formulation of sequential quadratic programming (SQP) methods for large-scale nonlinear optimization. SQP methods find an approximate solution of a sequence of quadratic programming (QP) subproblems in which a quadratic model of the objective function is minimized subject to the linearized constraints. Extensive numerical results are given for 1153 problems from the `CUTEst` test collection. The results indicate that SQP methods based on maintaining a quasi-Newton approximation to the Hessian of the Lagrangian function are both reliable and efficient for general large-scale optimization problems. In particular, the results show that in some situations, quasi-Newton methods are more efficient than competing methods based on the exact Hessian of the Lagrangian. The paper concludes with discussion of an SQP method that employs both approximate and exact Hessian information. In this approach the quadratic programming subproblem is either the conventional subproblem defined in terms of a positive-definite quasi-Newton approximate Hessian, or a convexified problem based on the exact Hessian.

Key words. nonlinear programming, nonlinear inequality constraints, sequential quadratic programming, second-derivative methods

AMS subject classifications. 49J20, 49J15, 49M37, 49D37, 65F05, 65K05, 90C30

1. Introduction.

This paper concerns the formulation of a sequential quadratic programming (SQP) method for the solution of the nonlinear optimization problem

$$\begin{aligned} &\underset{x\in\mathbb{R}^n}{\text{minimize}} && f(x) \\ &\text{subject to} && \ell \leq \begin{pmatrix} x \\ Ax \\ c(x) \end{pmatrix} \leq u, \end{aligned} \tag{1.1}$$

where $f(x)$ is a linear or nonlinear objective function, $c(x)$ is a vector of m nonlinear constraint functions $c_i(x)$, A is a matrix, and ℓ and u are vectors of lower and upper bounds. For simplicity, we assume in Sections 1–2 that the problem has the form

$$\text{(NP)} \qquad \underset{x\in\mathbb{R}^n}{\text{minimize}} \;\; f(x) \quad \text{subject to} \quad c(x) \geq 0, \tag{NP}$$

where f and the m components of the constraint vector c are assumed to be twice continuously differentiable for all $x \in \mathbb{R}^n$. Any linear constraints and simple bound constraints are included in the definition of c. (However, we emphasize that the exploitation of the properties of linear constraints is an important issue in the solution of large-scale problems.)

No assumptions are made about f and c (other than twice differentiability), which implies that the problem need not be convex. The vector $g(x)$ denotes the gradient of

*Department of Mathematics, University of California, San Diego, La Jolla, CA 92093-0112 (`pgill@ucsd.edu`, `elwong@ucsd.edu`). Research supported in part by Northrop Grumman Aerospace Systems, and National Science Foundation grants DMS-1318480 and DMS-1361421

†Systems Optimization Laboratory, Department of Management Science and Engineering, Stanford University, Stanford, CA 94305-4121 (`saunders@stanford.edu`). Research supported in part by the National Institute of General Medical Sciences of the National Institutes of Health [award U01GM102098]. The content is solely the responsibility of the authors and does not necessarily represent the official views of the funding agencies.

$f(x)$, and $J(x)$ denotes the $m \times n$ constraint Jacobian, which has ith row $\nabla c_i(x)^T$, the gradient of the ith constraint function $c_i(x)$. The Lagrangian associated with (NP) is $L(x, y) = f(x) - c(x)^T y$, where y is the m-vector of dual variables associated with the inequality constraints $c(x) \geq 0$. The Hessian of the Lagrangian with respect to x is denoted by $H(x, y) = \nabla^2 f(x) - \sum_{i=1}^{m} y_i \nabla^2 c_i(x)$.

Sequential quadratic programming methods find an approximate solution of a sequence of quadratic programming (QP) subproblems in which a quadratic model of the objective function is minimized subject to the linearized constraints. In a line-search SQP method, the QP solution provides a direction of improvement for a merit function that represents a compromise between the (usually conflicting) aims of minimizing the objective function and minimizing the constraint violations. Almost all SQP methods use an active-set quadratic programming method to solve the QP subproblem. In this situation, the SQP method has a major/minor iteration structure in which each minor iteration is an iteration of the active-set QP solver. The work for a minor iteration is dominated by the cost of solving a large system of symmetric indefinite linear equations that involve a subset of the variables and constraints.

A completely different approach to handling the inequality constraints of problem (NP) is to use an interior-point method. Interior methods follow a continuous path that passes through a solution of (NP). In the simplest case, the path is parameterized by a positive scalar parameter μ that may be interpreted as a perturbation for the first-order optimality conditions for the problem (NP). As $\mu \to 0$, a point $x(\mu)$ on the path satisfies $x(\mu) \to x^*$, where x^* is a solution of (NP). Each point on the path may be found by applying Newton's method to a system of nonlinear equations that represents perturbed optimality conditions for the original problem (NP). Each iteration of Newton's method involves a system of linear equations defined in terms of the derivatives of f and c. The Newton equations may be written in symmetric form, in which case each iteration requires the solution of a single symmetric indefinite system of equations involving the derivatives of every constraint in the problem.

The conventional wisdom is that when solving a general nonlinear problem "from scratch" (i.e., with no prior knowledge of the properties of a solution), software based on an interior method is generally faster and more reliable than software based on an SQP method. However, as SQP methods have the potential of being able to capitalize on a good initial starting point, they are considered to be more effective for solving a sequence of similar problems, such as a sequence of discretized continuous problems for which some underlying discretization is being refined. This claim is difficult to verify, however, as most test collections include unrelated problems of varying sizes and difficulty, or groups of problems with similar characteristics but slightly different formulations. Providing a fair comparison of SQP and interior methods is also complicated by the fact that very few SQP software packages are able to exploit the second derivatives of a problem. (This issue is considered further in Section 2.) Moreover, interior methods are more straightforward to implement with second derivatives, and most facilities for performing battery tests of optimization software provide test problems for which second derivatives are available automatically. Unfortunately, there are many practical problems for which even *first* derivatives are difficult or expensive to compute, let alone second derivatives. Test results from second-derivative methods are unlikely to be representative in this case.

The purpose of this paper is twofold. First, we provide a comparison of two widely-used software packages for general nonlinear optimization, one an interior method (`IPOPT`) and one an SQP method (`SNOPT7`). These packages are applied to almost all

the problems from the CUTEst testing environment. The tests are formulated so that the same derivative information is provided to both packages. In this environment it is shown that conclusions concerning the relative performance of first-derivative interior and SQP are more nuanced than the conventional wisdom. In particular, it is shown that active-set methods can be efficient for the solution of "one-off" problems, as is the case with active-set methods for linear programming. In other words, SQP methods may be best suited for some problems, and interior methods may be best for others.

If software is intended to be used in an environment in which second derivatives are available, then it is clear that the method that can best exploit these derivatives should be used. The second purpose of this paper is to extend conventional SQP methods so that second-derivatives can be exploited reliably and efficiently when they are available. These extensions are motivated by some comparisons of first-derivative SQP methods with second-derivative interior methods.

2. Background on SQP methods. The two principal ingredients of an SQP method for constrained optimization are: (i) a scalar-valued merit function $\mathcal{M}$ that provides a measure of the quality of a given point as an estimate of a solution of the constrained problem; and (ii) a direction of improvement for $\mathcal{M}$ defined as the solution of a quadratic programming subproblem. As in the unconstrained case, the merit function is used in conjunction with a line-search model to define a sufficient decrease in $\mathcal{M}$ at each iteration. Here we focus on the formulation and solution of the QP subproblem. For more background on the properties of SQP methods see, e.g., Gill and Wong [19].

2.1. Properties of the QP subproblem. Given the kth estimate (x_k, y_k) of the primal and dual solution, a conventional SQP method defines a direction $p_k = \widehat{x}_k - x_k$, where $\widehat{x}_k$ is a solution of the QP subproblem

$$\begin{aligned} \underset{x\in\mathbb{R}^n}{\text{minimize}} \quad & g(x_k)^T(x-x_k) + \tfrac{1}{2}(x-x_k)^T\widehat{H}_k(x-x_k) \\ \text{subject to} \quad & J(x_k)(x-x_k) \geq -c(x_k), \end{aligned} \tag{2.1}$$

where $\widehat{H}_k$ is an exact or approximate Hessian of the Lagrangian, and the QP objective function is a local quadratic model of $f(x)$ on the surface $c(x) - s = c(x_k) - s_k$, where $s_k = \max\{c(x_k), 0\}$. If the QP subproblem (2.1) has a bounded solution, then the first-order optimality conditions imply the existence of a primal-dual pair $(\widehat{x}_k, \widehat{y}_k)$ such that

$$g(x_k) + \widehat{H}_k(\widehat{x}_k - x_k) = J(x_k)^T\widehat{y}_k, \qquad \widehat{y}_k \geq 0, \tag{2.2}$$

$$r(\widehat{x}_k) \cdot \widehat{y}_k = 0, \qquad r(\widehat{x}_k) \geq 0, \tag{2.3}$$

where $r(x)$ is the vector of constraint residuals $r(x) = c(x_k) + J(x_k)(x - x_k)$, and $a \cdot b$ denotes the vector with ith component $a_i b_i$. At any feasible point x, the active set associated with the QP subproblem is given by

$$\mathcal{A}(x) = \{\, i : r_i(x) = [c(x_k) + J(x_k)(x - x_k)]_i = 0 \,\}.$$

The optimality conditions (2.1) may be characterized in terms of an index set $\mathcal{W} \subseteq \mathcal{A}(\widehat{x}_k)$ such that the rows of $J(x_k)$ with indices in $\mathcal{W}$ are linearly independent. It can be shown that conditions (2.2)–(2.3) may be written in the form

$$\begin{aligned} g(x_k) + \widehat{H}_k(\widehat{x}_k - x_k) &= J_w(x_k)^T\widehat{y}_w, \qquad \widehat{y}_w \geq 0, \\ c_w(x_k) + J_w(x_k)(\widehat{x}_k - x_k) &= 0, \qquad\qquad r(\widehat{x}_k) \geq 0, \end{aligned}$$

where $c_w(x_k)$ and $J_w(x_k)$ denote the rows of $c(x_k)$ and $J(x_k)$ associated with indices in $\mathcal{W}$. The vector $\widehat{y}_w$ is the subvector of $\widehat{y}_k$ such that $[\widehat{y}_k]_i = 0$ for $i \notin \mathcal{W}$, and $[\widehat{y}_k]_w = \widehat{y}_w$. These conditions may be written in matrix form

$$\begin{pmatrix} \widehat{H}_k & J_w(x_k)^T \\ J_w(x_k) & 0 \end{pmatrix} \begin{pmatrix} p_k \\ -\widehat{y}_w \end{pmatrix} = - \begin{pmatrix} g(x_k) \\ c_w(x_k) \end{pmatrix}, \tag{2.4}$$

where $p_k = \widehat{x}_k - x_k$. The working set $\mathcal{W}$ is said to be *second-order consistent* with respect to $\widehat{H}_k$ if the reduced Hessian $Z_w^T \widehat{H}_k Z_w$ is positive definite, where the columns of Z_w form a basis for the null-space of $J_w(x_k)$. If $\mathcal{W}$ is second-order consistent with respect to $\widehat{H}_k$, then system (2.4) is nonsingular and defines unique vectors p_k and $\widehat{y}_w$ satisfying

$$p_k^T \widehat{H}_k p_k = -\big(g(x_k) - J_w(x_k)^T \widehat{y}_w\big)^T p_k = -g_{\scriptscriptstyle L}(x_k, \widehat{y}_k)^T p_k, \tag{2.5}$$

where $g_{\scriptscriptstyle L}(x, y)$ denotes the gradient of the Lagrangian function with respect to x, i.e., $g_{\scriptscriptstyle L}(x, y) = g(x) - J(x)^T y$.

2.2. A nonbinding active-set method for solving the QP subproblem. To simplify exposition of the concurrent convexification algorithm in Section 5.1, we consider a generic QP of the form

$$\begin{array}{ll} \underset{x \in \mathbb{R}^n}{\text{minimize}} & \varphi(x) = g^T(x - x_{\scriptscriptstyle I}) + \frac{1}{2}(x - x_{\scriptscriptstyle I})^T H (x - x_{\scriptscriptstyle I}) \\ \text{subject to} & Ax \geq Ax_{\scriptscriptstyle I} - b, \end{array} \tag{2.6}$$

where $x_{\scriptscriptstyle I}$, b, A, g, and H are constant. In the SQP context, $x_{\scriptscriptstyle I} = x_k$, $g = g(x_k)$, $b = c(x_k)$, $A = J(x_k)$, and H is the exact or approximate Hessian of the Lagrangian. Thus, the objective is not necessarily convex and the QP subproblem may be indefinite.

The method described by Gill and Wong in [20] and implemented in the software package `SQIC` [21] is a two-phase method for general (i.e., possibly nonconvex) QP. In the first phase, the objective function is ignored while a conventional phase-one linear program is used to find a feasible point x_0 for the constraints $Ax \geq Ax_{\scriptscriptstyle I} - b$. On completion of the first phase, a working set $\mathcal{W}_0$ is available that contains the indices of a linearly independent subset of the constraints that are active at x_0. If A_0 denotes the $m_0 \times n$ matrix of rows of A with indices in $\mathcal{W}_0$, then

$$A_0 x_0 = A_0 x_{\scriptscriptstyle I} - b_0.$$

In the second phase, a sequence of primal-dual iterates $\{(x_j, y_j)\}_{j \geq 0}$ and linearly independent working sets $\{\mathcal{W}_j\}$ are generated such that: (i) $\{x_j\}_{j \geq 0}$ is feasible; (ii) $\varphi(x_j) \leq \varphi(x_{j-1})$; and (iii) for every $j \geq 1$, (x_j, y_j) is the primal and dual solution of the equality constrained problem defined by minimizing $\varphi(x)$ on the working set $\mathcal{W}_j$. The vector x_j associated with the primal-dual pair (x_j, y_j) is known as a *subspace minimizer* with respect to $\mathcal{W}_j$. If A_j denotes the $m_j \times n$ matrix of rows of A with indices in $\mathcal{W}_j$, then a subspace minimizer is formally defined as the point x_j such that $g(x_j) = A_j^T y_j$, and the KKT matrix

$$K_j = \begin{pmatrix} H & A_j^T \\ A_j & 0 \end{pmatrix} \tag{2.7}$$

has m_j negative eigenvalues. Equivalently, the associated reduced Hessian $Z_j^T H Z_j$, where the columns of Z_j form a basis for the null-space of A_j, is positive definite. Thus,

for any K_j satisfying this property, the working set $\mathcal{W}_j$ is *second-order consistent with respect to* H.

In general, the first iterate x_0 will not minimize $\varphi(x)$ on $\mathcal{W}_0$, and one or more preliminary iterations are needed to find the first subspace minimizer x_1. An estimate of x_1 is defined by solving the equality-constrained QP subproblem

$$\underset{x}{\text{minimize}} \;\; \varphi(x) \quad \text{subject to} \;\; A_0(x - x_I) + b_0 = 0. \tag{2.8}$$

If the KKT matrix K_0 is second-order consistent, then the solution of this subproblem is given by $x_0 + p_0$, where p_0 satisfies the nonsingular system

$$\begin{pmatrix} H & A_0^T \\ A_0 & 0 \end{pmatrix} \begin{pmatrix} p_0 \\ -\widehat{y}_0 \end{pmatrix} = - \begin{pmatrix} g(x_0) \\ b_0 + A_0(x_0 - x_I) \end{pmatrix} = - \begin{pmatrix} g(x_0) \\ 0 \end{pmatrix}. \tag{2.9}$$

If $x_0 + p_0$ is feasible, then $(x_1, y_1) = (x_0 + p_0, \widehat{y}_0)$; otherwise one of the constraints violated at $x_0 + p_0$ is added to the working set and the iteration is repeated. Eventually, the working set will include enough constraints to define an appropriate primal-dual pair (x_1, y_1).

If the first subspace minimizer x_1 is not optimal, then the method proceeds to find the sequence of subspace minimizers x_2, x_3, ..., described above. At any given iteration, not all the constraints in $\mathcal{W}_j$ are necessarily active at x_j. If every working-set constraint *is* active, then $\mathcal{W}_j \subseteq \mathcal{A}(x_j)$, and x_j is called a *standard* subspace minimizer; otherwise x_j is a *nonstandard* subspace minimizer. The method is formulated so that there is a subsequence of "standard" iterates intermixed with a finite number of consecutive "nonstandard" iterates. If the multipliers y_j are nonnegative at a standard iterate, then x_j is optimal for (2.6) and the algorithm is terminated. Otherwise, a working set constraint with a negative multiplier is identified and designated as the *nonbinding working-set constraint* associated with the subsequent consecutive sequence of nonstandard iterates. If the index of the nonbinding constraint corresponds to row s of A, then $[y_j]_s < 0$. There follows a sequence of "intermediate" iterations in which the constraint $a_s^T x \geq a_s^T x_I - b_s$ remains in the working set, though it is no longer active, while its multiplier is driven to zero. At each of these iterations, a search direction is defined by solving the equality-constrained subproblem

$$\underset{p \in \mathbb{R}^n}{\text{minimize}} \;\; \varphi(x_j + p) \quad \text{subject to} \;\; a_i^T p = \begin{cases} 0 & \text{if } i \neq s,\ i \in \mathcal{W}_j, \\ 1 & \text{if } i = s. \end{cases} \tag{2.10}$$

In matrix form, the optimality conditions for subproblem (2.10) are

$$\begin{pmatrix} H & A_j^T \\ A_j & 0 \end{pmatrix} \begin{pmatrix} p_j \\ -q_j \end{pmatrix} = \begin{pmatrix} 0 \\ e_s \end{pmatrix}, \tag{2.11}$$

where $y_j + q_j$ are the multipliers at the minimizer $x_j + p_j$, and e_s denotes the sth column of the identity matrix. (To simplify the notation, it is assumed that the nonbinding constraint corresponds to the sth row of A, which implies that a_s^T is the sth row of both A and A_j.) Any nonzero step along p_j increases the residual of the nonbinding constraint while maintaining the residuals of the other working-set constraints at zero (i.e., the nonbinding constraint becomes inactive while the other working-set constraints remain active).

Once the direction (p_j, q_j) has been computed, the computation of the next iterate x_{j+1} depends on the value of $p_j^T H p_j$, the curvature of φ along p_j. There are two cases to consider.

Case 1: $p_j^T H p_j > 0$. In this case the curvature is positive along p_j. This will always be the outcome when φ is convex. In this case, the step to the minimizer of φ along the search direction p_j is given by

$$\alpha_j^* = -g(x_j)^T p_j / p_j^T H p_j = -[y_j]_s / p_j^T H p_j. \tag{2.12}$$

The definition of α_j^* implies that the multiplier $[y_j + \alpha_j^* q_j]_s$ associated with the nonbinding constraint at $x_j + \alpha_j^* p_j$ is zero. This implies that if $x_j + \alpha_j^* p_j$ is feasible with respect to the constraints that are not in the working set, then the nonbinding constraint index can be removed from $\mathcal{W}_j$ as the conditions for a subspace minimizer continue to hold. This gives a new standard iterate $x_{j+1} = x_j + \alpha_j^* p_j$, with working set $\mathcal{W}_{j+1} = \mathcal{W}_j \setminus \{s\}$. Either x_{j+1} is optimal or a new nonbinding constraint is identified and the process is repeated. If $x_j + \alpha_j^* p_j$ is not feasible, then x_{j+1} is defined as $x_j + \alpha_j p_j$, where α_j is the largest step that gives a feasible $x_j + \alpha_j p_j$. The point x_{j+1} must have at least one constraint that is active but not in $\mathcal{W}_j$. If t is the index of this constraint, and a_t and the vectors $\{a_i\}_{i\in\mathcal{W}_j}$ are linearly independent, then t is added to the working set to give $\mathcal{W}_{j+1}$. At the next iteration, a new value of (p_j, q_j) is computed using system (2.11) defined with A_{j+1}. If a_t and $\{a_i\}_{i\in\mathcal{W}_j}$ are linearly dependent, then it is shown in [20] that the working set $\mathcal{W}_{j+1} = \{\mathcal{W}_j \setminus \{s\}\} \cup \{t\}$ defined by replacing the index t with index s is linearly independent. Moreover, $x_{j+1} = x_j + \alpha_j p_j$ is a subspace minimizer with respect to $\mathcal{W}_{j+1}$.

Case 2: $p_j^T H p_j \leq 0$. In this case H is not positive definite and $\varphi(x_j + \alpha p_j)$ is unbounded below for positive values of α. Either the QP is unbounded, or there exists a constraint index t and a nonnegative step $\widehat{\alpha}_j$ such that the constraint residuals satisfy $r_t(x_j + \widehat{\alpha}_j p_j) = 0$, $r(x_j + \widehat{\alpha}_j p_j) \geq 0$, and $\widehat{\alpha}_j$ minimizes $\varphi(x_j + \alpha p_j)$ for all feasible $x_j + \alpha p_j$. In this case, $x_{j+1} = x_j + \widehat{\alpha}_j p_j$ and, as above, either a_t and $\{a_i\}_{i\in\mathcal{W}_j}$ are linearly independent, in which case $\mathcal{W}_{j+1} = \mathcal{W}_j \cup \{t\}$, or the working set defined by replacing the index t with index s, is linearly independent. Moreover, $x_{j+1} = x_j + \alpha_j p_j$ is a subspace minimizer with respect to $\mathcal{W}_{j+1}$.

In both cases, the process is repeated at the next subspace minimizer defined by an appropriate working set until an optimal solution is found or the problem is declared to be unbounded.

2.3. Difficulties associated with using second derivatives. If $H(x_k, y_k)$ is positive definite and the QP subproblem Hessian is $\widehat{H}_k = H(x_k, y_k)$, then the inequality $g_L(x_k, \widehat{y}_k)^T p_k < 0$ holds and the SQP search direction is a descent direction for the Lagrangian defined with multipliers $y = \widehat{y}_k$. The curvature condition $p_k^T H(x_k, y_k) p_k > 0$ is sufficient for the existence of a step length that provides a sufficient decrease for several merit functions that have been proposed in the literature; e.g., the ℓ_1 penalty function (Han [31] and Powell [34]) and various forms of the augmented Lagrangian merit function (Han [31], Schittkowski [35], and Gill, Murray, Saunders and Wright [16]).

If problem (NP) is not convex, the Hessian of the Lagrangian may be indefinite, even in the neighborhood of a solution. This situation creates a number of difficulties in the formulation and analysis of a conventional SQP method.

(i) In the nonconvex case, QP subproblem (2.1) may be nonconvex, which implies that the objective of (2.1) may be unbounded below in the feasible region, and that there may be many local solutions. In addition, nonconvex QP is NP-hard—even for the calculation of a local minimizer [3, 9]. The complexity of the QP subproblem has been a major impediment to the formulation of second-

derivative SQP methods (although methods based on indefinite QP subproblems have been proposed [5, 6]).

(ii) If $H(x_k, y_k)$ is not positive definite, then p_k may not be a descent direction for the merit function. This implies that an alternative direction must be found or the line search must allow the merit function to increase on some iterations (see, e.g., Grippo, Lampariello and Lucidi [28, 29, 30], Toint [39], and Zhang and Hager [44]).

Over the years, algorithm developers have avoided these difficulties by solving a convex QP subproblem defined with a positive semidefinite quasi-Newton approximate Hessian. In this form, SQP methods have proved reliable and efficient for many problems. For example, under mild conditions the general-purpose solvers `NLPQL` [36], `NPSOL` [15, 16], `DONLP` [38], and `SNOPT7` [13] typically find a (local) optimum from an arbitrary starting point, and they require relatively few evaluations of the problem functions and gradients.

3. Numerical results. Before addressing the use of second-derivatives in SQP methods, we present numerical results that have motivated our work. This section includes a summary of the results obtained by running the SQP package `SNOPT7` [13] and the interior-point package `IPOPT` [43, 40, 42] on problems in the `CUTEst` test collection [23]. `IPOPT` is arguably the most successful and widely-used package for nonlinearly constrained optimization. The version of `IPOPT` used in the runs was version 3.11.8, running with linear solver MA57.

`SNOPT7` version 7.4 is a Fortran implementation of the general sequential quadratic programming method discussed in Section 1. `SNOPT7` is designed to solve large-scale problems of the form (1.1). Internally, `SNOPT7` transforms this problem into standard form by introducing a vector of slack variables s. The equivalent problem is

$$\underset{x,s}{\text{minimize}} \;\; f(x) \quad \text{subject to} \;\; \begin{pmatrix} Ax \\ c(x) \end{pmatrix} - s = 0, \quad l \leq \begin{pmatrix} x \\ s \end{pmatrix} \leq u. \tag{3.1}$$

All runs were made on a MacPro configured with a 2.7GHz 12-core Intel Xeon E5 processor and 64GB of RAM. `SNOPT7` was compiled using gfortran 4.6 with full code optimization and the optimized BLAS library in the Accelerate framework from Apple. The floating-point precision was 2.22×10^{-16}.

3.1. The active-set method of `SNOPT7`. To solve the QP subproblems, `SNOPT7` employs the convex QP solver `SQOPT` [14], which is an implementation of a reduced-Hessian, reduced-gradient active-set method. As `SNOPT7` was designed prior to the development of the proposed concurrent convexification algorithm, the active-set method in `SQOPT` differs from the one described in Section 2.2.

The method of `SQOPT` is a binding active-set method with the computed QP search directions lying in the null-space of the constraints in the working set. A constraint from the working set is removed at the *beginning* of a sequence of intermediate iterates, whereas the nonbinding method of Section 2.2 removes the constraint at the end of the sequence. For a binding active-set method, the reduced Hessian Z^THZ is positive semidefinite with at most one zero eigenvalue. If the reduced Hessian is positive definite, a suitable direction is computed from the system

$$Z^THZp_S = -Z^Tg \tag{3.2}$$

using a dense Cholesky factor of Z^THZ. If the reduced Hessian is singular, the Cholesky factor is used to define p_S such that $Z^THZp_S = 0$ and $p_S^TZ^Tg < 0$.

TABLE 1
The 1156 *CUTEst problems listed by frequency and type.*

Type	Frequency	Characteristics
LP	26	Linear obj, linear constraints
QP	238	Quadratic obj, linear constraints
UC	173	Nonlinear obj, no constraints
BC	141	Nonlinear obj, bound constraints
LC	70	Nonlinear obj, linear constraints
NC	408	Nonlinear obj, nonlinear constraints
FP	100	Constant objective function
NS	19	Non-smooth

If a binding-direction method is applied to the QP subproblem associated with (3.1), then all the inequality constraints are simple bounds and the number of degrees of freedom is the same as the number of superbasic variables (see Gill, Murray and Saunders [13]). The size of the reduced Hessian is equal to the number of superbasic variables for the current working set. If this number is large, then solving (3.2) becomes expensive.

3.2. The CUTEst test collection. The CUTEst distribution of January 14, 2015 (Subversion revision 245) contains 1156 problems in standard interface format (SIF). A list of CUTEst problem types and their frequency is given in Table 1. Although many problems allow for the number of variables and constraints to be adjusted in the SIF file, all the tests are based on problem dimensions set in the CUTEst distribution. The three problems *recipe*, *s365*, and *s365mod* are omitted because of undefined variables or floating-point exceptions in the SIF file. This defines a grand total of 1153 problems ranging in size from *hs1* (two variables and no constraints) to *bdry2* (251001 variables and 250498 constraints). Of these 1153 problems attempted, 137 have more than 3000 degrees of freedom, with the largest nonlinearly constrained problem (*jannson3*) having almost 20000 degrees of freedom at the solution.

The 19 problems *bigbank*, *bridgend*, *britgas*, *concon*, *core1*, *core2*, *gridgena*, *hs67*, *hs85*, *hs87*, *mconcon*, *net1*, *net2*, *net3*, *net4*, *stancmin*, *twiribg1*, *twirimd1*, and *twirism1* are non-smooth, but are included in the test-set nevertheless. The 5 problems *fletchbv*, *indef*, *lukvle2*, *mesh*, and *static3* are known to have an objective function that is unbounded below in the feasible region.

Many of the problems are either infeasible or have no known feasible point. The 14 problems *a2nndnil*, *a5nndnil*, *arglale*, *arglble*, *arglcle*, *flosp2hh*, *flosp2hl*, *flosp2hm*, *ktmodel*, *lincont*, *model*, *nash*, *synpop24*, and *woodsne* have infeasible linear constraints. No method for finding a local solution of an optimization problem with nonlinear constraints is assured of finding a feasible point. The failure of an algorithm to find a feasible point in this situation does not imply that the problem is infeasible. The two nonlinear problems *burkehan* and *flosp2th* are known to be infeasible. Another 17 problems have no known feasible point: *argauss*, *arwhdne*, *cont6-qq*, *drcavty3*, *eigenb*, *growth*, *hadamard*, *himmelbd*, *junkturn*, *lewispol*, *lubrifc*, *lukvle16*, *nuffield*, *nystrom5*, *powellsq*, *tro21x5*, and *tro41x9*. We conjecture that these problems are infeasible. (Problems *junkturn* and *nystrom5* are feasible if the constraints are perturbed by 10^{-3}.)

3.3. User-specified options. Both SNOPT7 and IPOPT allow the user to replace the values of certain default run-time options. With the exception of an 1800-second time-limit, all IPOPT runs were made using the default options. Figure 1 lists the

```
BEGIN SNOPT Problem                          #BEGIN IPOPT Problem
  Superbasics limit           150000           max_iter              10000
  Reduced Hessian dimension     4000           # next option for L-BFGS only
  Major iterations             10000           hessian_approximation limited-memory
  Iteration limit             100000           linear_solver         ma57
  Major optimality  tolerance 1.22e-4          time_limit            1800
  Time limit                   1800          #END IPOPT Problem
END SNOPT Problem
```

FIG. 1. *The SNOPT7 and IPOPT run-time option files.*

SNOPT7 and IPOPT options that differ from their default values. (For a complete list of these options see [43] and [13].) `Reduced Hessian dimension` specifies the maximum size of the dense reduced Hessian available for SQOPT. If the number of degrees of freedom exceeds this value during the QP solution, SQOPT solves a perturbed version of (3.2) using the conjugate-gradient-type solver SYMMLQ [33]. The default `Major optimality` tolerance for SNOPT7 is 2×10^{-6} (see Section 2.11 of Gill, Murray and Saunders [13]) The larger value of 1.22×10^{-4} was used to match the default optimality tolerance of IPOPT.

3.4. Results obtained using first derivatives only. SNOPT7 and IPOPT with the L-BFGS option were applied to the 1153 test problems listed in Table 1. The results are summarized in Table 2. The constraint format (3.1) allows SNOPT7 to find a feasible point for the linear constraints before evaluating the objective function and nonlinear constraints. If the linear constraints are infeasible, SNOPT7 terminates immediately without computing the nonlinear functions. Otherwise, all subsequent major iterates satisfy the linear constraints. (Sometimes this feature helps ensure that the functions and gradients are only computed at points where they are well defined.) For brevity, Table 2 lists the number of infeasible problems found by SNOPT7 without distinguishing between linear and nonlinear constraints.

TABLE 2
SNOPT7 and first-derivative IPOPT on 1153 CUTEst problems.

SNOPT7	
Optimal	994
Optimal, but low accuracy	8
Unbounded	5
Infeasible constraints	16
Locally infeasible constraints	16
Total successes	1039
False infeasibility	19
Iteration limit	56
Time limit	15
Numerical problems	16
Final point cannot be improved	8
Total failures	114

IPOPT	
Optimal	768
Optimal, but low accuracy	147
Unbounded	2
Infeasible constraints	9
Locally infeasible constraints	7
Total successes	933
False infeasibility	7
Iteration limit	77
Time limit	21
Diverges	9
Restoration failed	64
Too few degrees of freedom	33
Regularization too large	1
Invalid derivative entry	5
Search direction too small	2
Segmentation fault	1
Total failures	220

Of the 19 nonsmooth problems, SNOPT7 solved all but *gridgena*, *hs87*, and *net4*, and IPOPT solved all but *bigbank*, *gridgena*, *hs87*, and *net4*. The 5 unbounded problems: *fletchbv*, *indef*, *lukvle2*, *mesh*, and *static3* were identified correctly by SNOPT7.

IPOPT terminated 11 cases with an "unbounded" diagnostic message, but, of these, only *indef* and *static3* were identified correctly.

If any QP subproblem is infeasible, or the Lagrange multipliers of the subproblem become large, then SNOPT7 switches to "elastic mode". In this mode, the nonlinear constraint functions are allowed to violate their bounds by an amount that is multiplied by a positive weight and included in the objective function (see, e.g., Gill, Murray and Saunders [13]). This feature allows SNOPT7 to find a local minimizer of the sum of infeasibilities if the nonlinear constraints appear to be infeasible. As mentioned above, the calculation of such a point does imply the problem is infeasible. A run was considered to have "failed" if a final point of local infeasibility was declared for a problem that is known to be feasible. SNOPT7 found "false" infeasible points for 19 problems: *discs*, *drugdis*, *eigmaxa*, *eigmina*, *fletcher*, *flosp2tm*, *flt*, *hs61*, *lootsma*, *lukvle17*, *lukvle18*, *mss2*, *mss3*, *s316-322*, *vanderm1*, *vanderm2*, *vanderm3*, *lukvli10*, and *pde2*. SNOPT7 solves problems *mss2* and *mss3* successfully when the optimality tolerance is reduced to the default value of 10^{-6}. Problems *fletcher* and *lootsma* have feasible solutions, but their initial points are infeasible and stationary for the sum of infeasibilities, and thus SNOPT7 terminated immediately. These problems are also listed as failures.

IPOPT with L-BFGS determined that 23 problems are infeasible. Seven of these (*cresc100*, *cresc50*, *ngone*, *pfit2*, *pfit4*, *wachbieg*, and *lukvli17*) have known feasible points and are included in the list of failures.

The results are summarized using performance profiles (in $\log_2$ scale) proposed by Dolan and Moré [4]. The idea of a performance profile is to provide an "at-a-glance" comparison of the performance of a set $\mathcal{S}$ of n_s solvers applied to a test set $\mathcal{P}$ of n_p problems. For each solver $s \in \mathcal{S}$ and problem $p \in \mathcal{P}$ in a profile, the number t_{ps} is the time (or some other measure, e.g., number of iterations) needed to solve problem p with solver s. To compare the performance of a problem p over the different solvers, the *performance ratio* for each successfully solved problem and solver is defined as

$$r_{ps} = \frac{t_{ps}}{\min\{t_{ps} : s \in \mathcal{S}\}}.$$

If r_{ms} denotes the maximum time needed over all problems that were solved successfully, then the performance ratio for problems that failed is defined as some value greater than r_{ms}.

Given the set of performance ratios, a function $P_s(\sigma)$ is defined for each solver such that

$$P_s(\sigma) = \frac{1}{n_p}|\{p \in \mathcal{P} : r_{ps} \leq \sigma\}|,$$

where $\sigma \in [1, r_{ms}]$. The value $P_s(\sigma)$ is the fraction of problems for solver s that were solved within σ of the best time. $P_s(1)$ is the fraction of problems for which s was the fastest solver. The value $P_s(r_{ms})$ gives the fraction of problems solved successfully by solver s.

The presented performance profiles are log-scaled, with $\tau = \log_2(\sigma)$ on the x-axis and the function

$$P_s(\tau) = \frac{1}{n_p}|\{p \in \mathcal{P} : \log_2(r_{ps}) \leq \tau\}|,$$

on the y-axis for each solver. The y-axis can be interpreted as the fraction of problems that were solved within 2^τ of the best time. Because the y-axis is the fraction of

problems solved, and the x-axis is the factor of time needed to solve a problem, the "best" solver should have a function $P_s(\tau)$ that lies towards the upper-left of the graph. If a problem is solved in 0.00 seconds, then that value is replaced by 0.001 to prevent division by zero in the calculation of the performance ratios.

Figure 2 gives the performance profiles for the solve times (in seconds) and total number function evaluations required by SNOPT7 and IPOPT on all 1153 problems. The left figure profiles the solve times, the right figure profiles function evaluations. In these profiles, an algorithm is considered to have solved a problem successfully if one of the first five outcomes listed in Table 2 occurs.

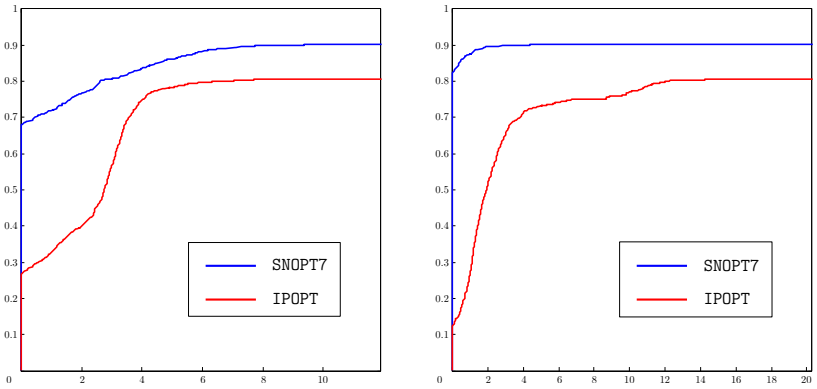

FIG. 2. *Performance profiles for SNOPT7 and IPOPT on 1153 CUTEst test problems using first derivatives. The IPOPT results were obtained using an L-BFGS approximate Hessian. The left figure profiles solve times (in seconds); the right figure profiles function evaluations.*

3.5. Comparison of IPOPT with second-derivatives and SNOPT7. In this section we consider results obtained by running SNOPT7 with first derivatives (the only option) and IPOPT with the second derivatives (the default option). The results are summarized in Table 3. Of the 19 nonsmooth problems, IPOPT solved all but *britgas*, *net2* and *net4*.

TABLE 3
SNOPT7 and second-derivative IPOPT on 1153 CUTEst problems.

SNOPT7	
Optimal	994
Optimal, but low accuracy	8
Unbounded	5
Infeasible constraints	16
Locally infeasible constraints	16
Total successes	1039
False infeasibility	19
Iteration limit	56
Time limit	15
Numerical problems	16
Final point cannot be improved	8
Total failures	114

IPOPT	
Optimal	984
Optimal, but low accuracy	14
Unbounded	1
Infeasible constraints	9
Locally infeasible constraints	7
Total successes	1015
False infeasibility	6
Iteration limit	46
Time limit	26
Too few degrees of freedom	33
Diverges	2
Restoration failed	17
Regularization too large	2
Invalid derivative entry	5
Search direction too small	1
Total failures	138

The second-derivative version of IPOPT determined that 21 problems are infeasible. Of these, the 6 problems *artif*, *brainpc2*, *cresc50*, *pfit1*, *pfit2*, and *wachbieg* have known feasible points and are included in the list of failures.

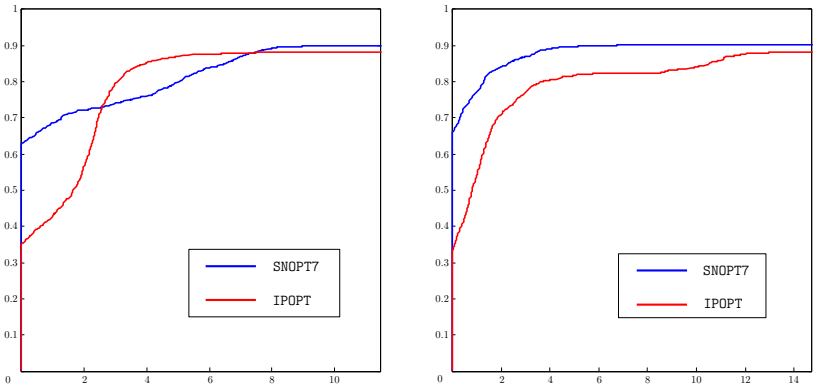

FIG. 3. *Performance profiles for SNOPT7 (first derivatives only) and second-derivative IPOPT on 1153 CUTEst test problems. The left figure profiles the solve times (in seconds), the right figure profiles function evaluations.*

Figure 3 gives the performance profiles for the solve times (in seconds) and total number of function evaluations required by SNOPT7 (first derivatives only) and second-derivative IPOPT on all 1153 problems. As above, an algorithm is considered to have solved a problem successfully if one of the first five outcomes listed in Table 3 occurs.

Two conclusions may be drawn from these results. First, IPOPT with the second-derivative option is significantly more robust than IPOPT with first derivatives only, which is why the IPOPT documentation strongly recommends the second-derivative option. Second, software based on a first-derivative SQP method can be competitive with software based on an interior method using second derivatives. The remaining discussion focuses upon the source of this competitiveness and how it may be exploited for the development of second-derivative SQP methods.

4. Second-derivative SQP methods. An important quantity that influences the efficiency of an optimization method is the number of degrees of freedom $n_{\rm df}$ at a solution. The 1153 problems in the CUTEst test collection include 1016 problems with $n_{\rm df} \leq 3000$ and 1095 problems with $n_{\rm df} \leq 4000$[1]. Generally speaking, methods that maintain an explicit reduced Hessian when solving the QP subproblem (such as the QP solver SQOPT used in SNOPT7) become less efficient as the number of degrees of freedom increases. Figure 4 illustrates how the efficiency of SNOPT7 is influenced by the number of degrees of freedom. Figure 4 profiles the solve times for SNOPT7 and IPOPT on the 1016 CUTEst test problems with $n_{\rm df} \leq 3000$. A comparison of the solution-time profiles of Figures 3 and 4 indicates that overall, SNOPT7 is more competitive on problems with few degrees of freedom at the solution. The IPOPT package uses a direct factorization of the KKT matrix, which implies that the efficiency is relatively unrelated to the number of degrees of freedom. It follows that as the number of degrees of freedom increases, the number of problems for which IPOPT has a faster solution time increases. For example, on the 68 problems with $n_{\rm df} > 4000$ only 24

[1]Here, the value of $n_{\rm df}$ is taken as the the number of degrees of freedom at a solution found by SNOPT7.

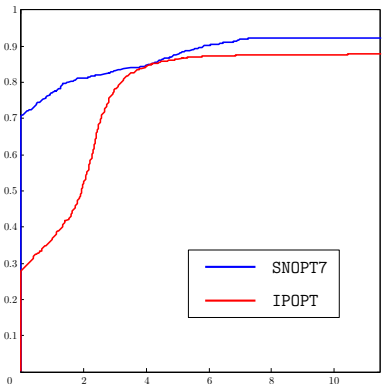

FIG. 4. *Performance profiles of solve times for* SNOPT7 *(first derivatives only) and second-derivative* IPOPT *on 1016 CUTEst test problems with no greater that 3000 degrees of freedom.*

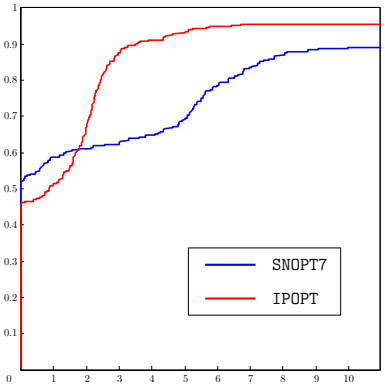

FIG. 5. *Performance profiles of solve times for* SNOPT7 *(first derivatives only) and second-derivative* IPOPT *on the 296 CUTEst test problems with either no constraints or only simple bounds.*

problems are solved faster with SNOPT7. These 68 problems provide a substantial test of the conjugate-gradient solver in SQOPT.

This inefficiency may be removed by using a QP solver that maintains an explicit reduced Hessian when the number of degrees of freedom is small, and uses direct factorization when the number of degrees of freedom is large. The QP-package SQIC [21] implements a method based on this strategy.

Given an efficient QP solver, once the QP working set settles down, the efficiency of SNOPT7 depends largely on whether or not the limited-memory method is able to adequately represent the Lagrangian Hessian. For example, many of the 68 large problems have no constraints or only simple bounds, and in these cases, the large number of major iterations is consistent with results obtained by other limited-memory quasi-Newton methods (see, e.g., [2, 11]). It is in this situation that the use of second derivatives, when available, can have a significant impact on the rate of convergence of the SQP method. Figure 5 profiles the solve times for SNOPT7 and IPOPT on 296 CUTEst test problems with either no constraints or only simple bounds.

The convexification method discussed in the next section provides the basis of an SQP method that first uses a quasi-Newton method to identify an estimate of the working set at a solution. This estimate is used to initiate a sequence of QP subproblems defined by the Hessian of the Lagrangian. Other methods that identify the active set using a convex QP based on a BFGS approximation of the Hessian have been proposed by Byrd et al. [1] and Gould and Robinson [24, 25, 26].

5. Convexification. Convexification is a process for defining a local convex approximation of a nonconvex problem. This approximation may be defined on the full space of variables or just on some subset. Many model-based optimization methods use some form of convexification. For example, line-search methods for unconstrained and linearly-constrained optimization define a convex local quadratic model in which the Hessian $H(x_k, y_k)$ is replaced by a positive-definite matrix $H(x_k, y_k) + E_k$ (see, e.g., Greenstadt [27], Gill and Murray [12], Schnabel and Eskow [37], and Forsgren and Murray [10]). All of these methods are based on convexifying an unconstrained or equality-constrained local model. Here we consider a method that convexifies the inequality-constrained subproblem directly. The method extends some approaches proposed by Gill and Robinson [17, Section 4] and Kungurtsev [32].

In the context of SQP methods, the purpose of the convexification is to find a primal-dual pair (x_k, y_k) and matrix ΔH_k such that

$$p_k^T \big(H(x_k, y_k) + \Delta H_k\big) p_k \geq \bar{\gamma} p_k^T p_k,$$

where $\bar{\gamma}$ is a fixed positive scalar that defines a minimum acceptable value of the curvature of the Lagrangian. Ideally, any algorithm for computing ΔH_k should satisfy two requirements. First, the convexification should be "minimally invasive", i.e., if $H(x_k, y_k)$ is positive definite or $p_k^T H(x_k, y_k) p_k \geq \bar{\gamma} p_k^T p_k$, then ΔH_k should be zero. Second, it must be possible to store the modification ΔH_k implicitly, without the need to modify the elements of $H(x_k, y_k)$.

The convexification discussed here can take three forms: preconvexification, concurrent convexification, and post-convexification. Not all of these modifications are necessary at a given iteration.

5.1. Concurrent QP convexification. Concurrent convexifcation is based on reformulating the active-set QP method of Section 2.2.

Assume that a QP search direction p_j with zero or negative curvature is detected after the selection of $a_s^T x \geq b_s$ as the nonbinding constraint (i.e., $[y_j]_s < 0$). In this case, H is not positive definite and the QP Hessian is modified so that it has sufficiently large positive curvature along p_j. As $p_j^T H p_j \leq 0$, the objective $\varphi(x_j + \alpha p_j)$ is unbounded below for positive values of α. In this case, either the unmodified QP is unbounded, or there exists a constraint index t and a nonnegative step $\widehat{\alpha}_j$ such that the constraint residuals satisfy $r_t(x_j + \widehat{\alpha}_j p_j) = 0$, $r(x_j + \widehat{\alpha}_j p_j) \geq 0$, and $\widehat{\alpha}_j$ minimizes $\varphi(x_j + \alpha p_j)$ for all feasible $x_j + \alpha p_j$.

If $p_j^T H p_j < 0$, the positive semidefinite rank-one matrix $\sigma a_s a_s^T$ is added to H implicitly. This modifies the quadratic program being solved, but the current iterate x_j remains a subspace minimizer for the modified problem. The only computed quantities altered by the modification are the curvature and the multiplier y_s associated with the nonbinding working-set constraint. The modified Hessian is defined as $H(\bar{\sigma}) = H + \bar{\sigma} a_s a_s^T$ for some $\bar{\sigma} > 0$. Gill and Wong [20] show that the curvature $p^T H p$ is nondecreasing during a sequence of nonstandard iterations associated with a

nonbinding index s. This implies that a modification of the Hessian will occur only at the first nonstandard iterate.

For an arbitrary σ, the gradient of the modified objective at x_j is

$$g + H(\sigma)(x_j - x_I) = g + (H + \sigma a_s a_s^T)(x_j - x_I).$$

As (x_j, y_j) is a standard subspace minimizer for the unmodified problem, the identities $g(x_j) = g + H(x_j - x_I) = A_j^T y_j$ and $a_s^T(x_j - x_I) = -b_s$ hold, and the gradient of the modified objective is given by

$$\begin{aligned} g + H(\sigma)(x_j - x_I) &= g + H(x_j - x_I) + \sigma a_s a_s^T (x_j - x_I) \\ &= g(x_j) + \sigma a_s^T (x_j - x_I) a_s \\ &= A_j^T \big(y_j - \sigma b_s e_s\big) = A_j^T y(\sigma), \quad \text{with} \quad y(\sigma) = y_j - \sigma b_s e_s. \end{aligned}$$

This implies that x_j is a subspace minimizer of the modified problem for all $\sigma \geq 0$. Moreover, the multipliers of the modified problem are the same as those of the unmodified problem except for the multiplier y_s associated with the nonbinding constraint, which is shifted by $-\sigma b_s$.

Once the Hessian is modified, the system (2.11) for the primal-dual direction becomes

$$\begin{pmatrix} H + \bar{\sigma} a_s a_s^T & A_j^T \\ A_j & 0 \end{pmatrix} \begin{pmatrix} \bar{p}_j \\ -\bar{q}_j \end{pmatrix} = \begin{pmatrix} 0 \\ e_s \end{pmatrix},$$

which is equivalent to

$$\begin{pmatrix} H & A_j^T \\ A_j & 0 \end{pmatrix} \begin{pmatrix} p_j \\ -(\bar{q}_j - \bar{\sigma} e_s) \end{pmatrix} = \begin{pmatrix} 0 \\ e_s \end{pmatrix}.$$

A comparison with (2.11) yields

$$\bar{p}_j = p_j \quad \text{and} \quad \bar{q}_j = q_j + \bar{\sigma} e_s,$$

which implies that the QP direction is not changed by the modification.

For any $\sigma \geq 0$, let $\alpha_j(\sigma)$ denote the step associated with the search direction for the modified QP. The identities $a_s^T p_j = 1$ and $a_s^T(x_j - x_I) = -b_s$ imply that

$$\begin{aligned} \alpha_j(\sigma) &= -\frac{\big(g + (H + \sigma a_s a_s^T)(x_j - x_I)\big)^T p_j}{p_j^T (H + \sigma a_s a_s^T) p_j} \\ &= -\frac{g(x_j)^T p_j + \sigma a_s^T (x_j - x_I)}{p^T H p + \sigma} \\ &= -\frac{g(x_j)^T p_j - \sigma b_s}{p^T H p + \sigma} = -\frac{y_s - \sigma b_s}{p^T H p + \sigma} = -\frac{y_s(\sigma)}{p^T H p + \sigma}. \end{aligned} \tag{5.1}$$

This implies that $\bar{\sigma}$ must be chosen large enough to satisfy

$$\bar{\sigma} > \sigma_{\min} = -p^T H p.$$

The derivative of $\alpha_j(\sigma)$ with respect to σ is given by

$$\alpha_j'(\sigma) = \frac{1}{(p_j^T H p_j + \sigma)^2} \Big(y_s + b_s p_j^T H p_j\Big) = \frac{y_s(\sigma_{\min})}{(p_j^T H p_j + \sigma)^2}. \tag{5.2}$$

The choice of $\bar{\sigma}$ depends on two parameters $y_{\texttt{tol}}$ and $d_{\max}$. The scalar $d_{\max}$ defines the maximum change in x at each QP iteration. The scalar $y_{\texttt{tol}}$ is the dual optimality tolerance and is used to define what is meant by a "nonoptimal" multiplier. In particular, the multiplier of the nonbinding constraint must satisfy $y_s < -y_{\texttt{tol}}$ in order to qualify as being nonoptimal.

There are two cases to consider for the choice of $\bar{\sigma}$.

Case (i): $b_s < 0$. In this case, $y_s(\sigma)$ is an increasing function of σ, which implies that there exists $\sigma_{\texttt{opt}} = (y_s - y_{\texttt{tol}})/b_s > 0$ such that $y_s(\sigma_{\texttt{opt}}) = y_{\texttt{tol}} > 0$. This modification changes the multiplier associated with the nonbinding constraint from nonoptimal to optimal. However, if $\sigma_{\texttt{opt}} < \sigma_{\min}$, then the curvature is not sufficiently positive and σ must be increased so that it is larger than $\sigma_{\texttt{opt}}$. The definition

$$\bar{\sigma} = \begin{cases} \sigma_{\texttt{opt}} & \text{if } \sigma_{\texttt{opt}} \geq 2\sigma_{\min}; \\ 2\sigma_{\min} & \text{if } \sigma_{\texttt{opt}} < 2\sigma_{\min}, \end{cases}$$

guarantees that the curvature along p_j is sufficiently positive with an optimal modified multiplier $y_s(\bar{\sigma})$. In either case, the QP algorithm proceeds by selecting an alternative nonbinding constraint without taking a step along p_j.

If $y_s(\sigma_{\min}) < 0$, it is possible to choose $\bar{\sigma} > \sigma_{\min}$ such that $y_s(\bar{\sigma})$ remains negative. The multiplier $y_s(\sigma)$ increases from the negative value $y_s(\sigma_{\min})$ to the value $-y_{\texttt{tol}}$ as σ increases from $\sigma_{\min}$ to the positive value $\sigma_{\texttt{nonopt}} = (y_s + y_{\texttt{tol}})/b_s$. This implies that if σ is chosen in the range $\sigma_{\min} < \sigma \leq \sigma_{\texttt{nonopt}}$, then the multiplier for the nonbinding constraint remains nonoptimal, and it is possible to both convexify and keep the current nonbinding constraint. However, in the SQP context it is unusual for a nonbinding constraint to have a negative value of b_s when x_k is far from a solution. For an SQP subproblem, b is the vector $c(x_k)$, and a negative value of b_s implies that the sth nonlinear constraint is violated at x_k. The linearization of a violated nonlinear constraint is likely to be retained in the working set because the SQP step is designed to reduce the nonlinear constraint violations. The picture changes when x_k is close a solution and the violations of the nonlinear constraints in the QP working set are small. In this case, if strict complementarity does not hold at the solution of the nonlinear problem and x_k is converging to a point that satisfies the second-order necessary conditions, but not the second-order sufficient conditions, then both b_s and y_s may be small and negative. It is for this reason that even if $y_s(\sigma_{\min})$ is negative, $\bar{\sigma}$ is chosen large enough that the multiplier changes sign and the nonbinding constraint is retained in the QP working set.

Case (ii): $b_s \geq 0$. In this case, $y_s(\sigma_{\min}) = y_s - b_s\sigma_{\min} < 0$ and $y_s(\sigma_{\min})$ decreases monotonically for all increasing $\sigma > \sigma_{\min}$. The step-length function $\alpha_j(\sigma)$ has a pole at $\sigma = -p^T H p$ and decreases monotonically, with $\alpha_j(\sigma) \to b_s \geq 0$ as $\sigma \to +\infty$. The behavior of $x(\sigma)$ is depicted in Figure 6 for a two-variable QP with constraints $a^T(x - x_I) \geq -b$, $x_1 \geq 0$, and $x_2 \geq 0$. The next iterate of the QP algorithm lies on the ray $x(\sigma) = x_j + \alpha_j(\sigma)p_j$. As $\sigma \to \infty$, $x(\sigma)$ moves closer to the point $x_j + b_s p_j$ on the hyperplane $a^T(x - x_I) = 0$.

A preliminary value of $\bar{\sigma}$ is chosen to give an x_{j+1} such that

$$\|x_{j+1} - x_j\|_2 \leq d_{\max},$$

where $d_{\max}$ is the preassigned maximum change in x at each QP iteration. If $\alpha_T = d_{\max}/\|p_j\|_2$, then the substitution of $\alpha_j(\bar{\sigma}) = \alpha_T$ in (5.1) gives the value $\bar{\sigma} = -(y_s + \alpha_T p_j^T H p_j)/(\alpha_T - b_s)$. However, the limit $\alpha_j(\sigma) \to b_s \geq 0$ as $\sigma \to +\infty$, implies that

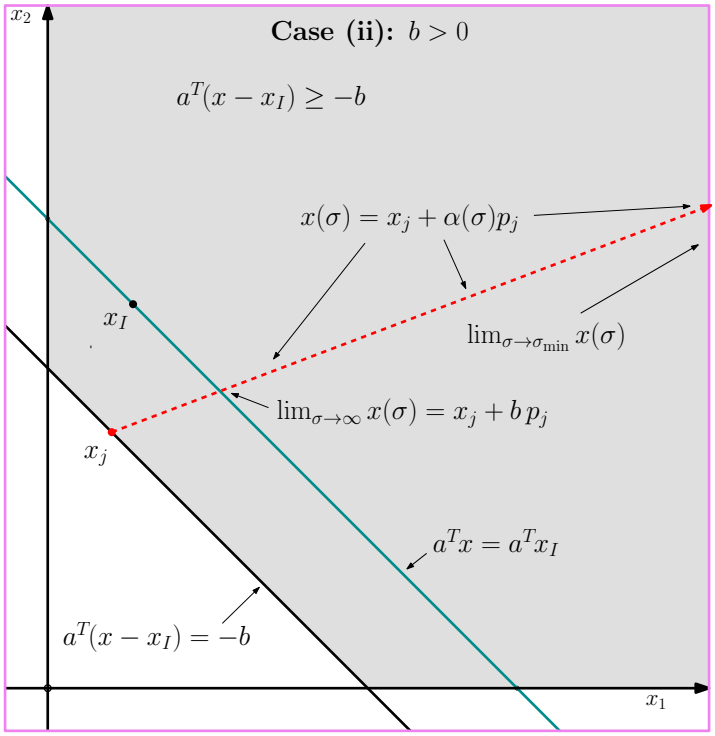

FIG. 6. *The figure depicts the feasible region for a QP with constraints* $a^T(x - x_I) \geq -b$, $x_1 \geq 0$, *and* $x_2 \geq 0$. *The point* x_j *is a standard subspace minimizer with working-set constraint* $a^T(x - x_I) \geq -b$. *The surface of the hyperplane* $a^T(x - x_I) = 0$ *is marked in green. The QP base point* x_I *is feasible for* $b \geq 0$. *The QP search direction is the red dotted line. The next iterate of the QP algorithm lies on the ray* $x(\sigma) = x_j + \alpha_j(\sigma)p_j$. *As the modification parameter* σ *increases from its initial value of* $\sigma_{\min}$, *the new iterate* $x(\sigma)$ *moves closer to the point* $x_j + b\,p_j$ *on the hyperplane* $a^T(x - x_I) = 0$.

this value of $\bar{\sigma}$ may be large if $\alpha_j(\bar{\sigma})$ is close to b_s. In order to avoid this difficulty, the value of $\bar{\sigma}$ is used as long as the associated value of $\alpha_j(\bar{\sigma})$ is sufficiently larger than b_s, i.e.,

$$\alpha_j(\bar{\sigma}) = \begin{cases} \alpha_T & \text{if } \alpha_T \geq 2b_s; \\ 2b_s & \text{if } \alpha_T < 2b_s, \end{cases} \quad \text{so that} \quad \bar{\sigma} = \begin{cases} -\dfrac{y_s + \alpha_T p_j^T H p_j}{\alpha_T - b_s} & \text{if } \alpha_T \geq 2b_s, \\ -\dfrac{y_s + 2b_s p_j^T H p_j}{b_s} & \text{if } \alpha_T < 2b_s. \end{cases}$$

If this algorithm is applied to a nonconvex QP of the form (2.6), then a solution is found for the convexified QP

$$\begin{aligned} \underset{x\in\mathbb{R}^n}{\text{minimize}} \quad & \varphi(x) = g^T(x - x_I) + \tfrac{1}{2}(x - x_I)^T(H + E)(x - x_I) \\ \text{subject to} \quad & Ax \geq Ax_I - b, \end{aligned} \tag{5.3}$$

where E is a positive-semidefinite matrix of the form $E = A^T\bar{\Sigma}A$, with $\bar{\Sigma}$ a positive

semidefinite diagonal matrix. In general, most of the diagonal elements of $\bar{\Sigma}$ are zero. The modification E may be reconstructed from A and a sparse representation of $\bar{\Sigma}$.

5.2. Preconvexification. The concurrent convexification method of Section 5.1 has the property that if x_0 is a subspace minimizer, then all subsequent iterates are subspace minimizers. Methods for finding an initial subspace minimizer utilize an initial estimate x_0 of the solution together with an initial working set $\mathcal{W}_0$ of linearly independent constraints. These estimates are often available from a phase-one linear program or, in the SQP context, the solution of the previous QP subproblem.

If a potential KKT matrix K_0 has too many negative or zero eigenvalues, then $\mathcal{W}_0$ is not a second-order consistent working set. In this case, an appropriate K_0 may be obtained by imposing temporary constraints that are deleted during the course of the subsequent QP iterations. For example, if n variables are temporarily fixed at their current values, then A_0 is the identity matrix and K_0 necessarily has exactly n negative eigenvalues regardless of the eigenvalues of $H(x_k, y_k)$. The form of the temporary constraints depends on the method used to solve the KKT equations; see, e.g., Gill and Wong [20, Section 6]. Once the temporary constraints are imposed, concurrent convexification can proceed as in Section 5.1 as the temporary constraints are removed from the working set during subsequent iterations.

A disadvantage of using temporary constraints is that it may be necessary to factor two KKT matrices if the initial working set is not second-order consistent. An alternative approach is to utilize the given working set $\mathcal{W}_0$ without modification and use *preconvexification*, which involves the definition of a positive-semidefinite E_0 such that the matrix

$$K_0 = \begin{pmatrix} H + E_0 & A_0^T \\ A_0 & 0 \end{pmatrix} \tag{5.4}$$

is second-order consistent. A suitable modification E_0 may be based on some variant of the symmetric indefinite or block-triangular factorizations of K_0. Appropriate methods include: (i) the inertia controlling LBL^T factorization (Forsgren [7], Forsgren and Gill [8]); (ii) an LBL^T factorization with pivot modification (Gould [22]); and (iii) a conventional LBL^T factorization of (5.4) with $E_0 = \sigma I$ for some nonnegative scalar σ (Wächter and Biegler [41]). In each case, the modification E_0 is zero if $\mathcal{W}_0$ is already second-order consistent.

5.3. Post-convexification. As concurrent convexification generates a sequence of second-order-consistent working sets, the SQP search direction $p_k = \widehat{x}_k - x_k$ must satisfy the second-order-consistent KKT system

$$\begin{pmatrix} H_k + E_k & J_w(x_k)^T \\ J_w(x_k) & 0 \end{pmatrix} \begin{pmatrix} p_k \\ -\widehat{y}_w \end{pmatrix} = - \begin{pmatrix} g(x_k) \\ c_w(x_k) \end{pmatrix}, \tag{5.5}$$

where $H_k = H(x_k, y_k)$ is the exact Hessian of the Lagrangian, E_k is the matrix defined by the pre- and/or concurrent convexification, and $c_w(x_k)$ and $J_w(x_k)$ are the rows of $c(x_k)$ and $J(x_k)$ associated with indices in the final QP working set $\mathcal{W}$ (cf. (2.4)). In most cases, concurrent convexification is sufficient to give $p_k^T(H_k + E_k)p_k > 0$, but it may hold that $p_k^T(H_k + E_k)p_k \leq 0$. In this case, p_k is not a descent direction for $g_L(x_k, \widehat{y}_k)$, and an additional *post-convexification step* is necessary. In the following discussion, there is no loss of generality in assuming that $E_k = 0$, i.e., it is assumed that H_k has not been modified during the preconvexification or concurrent convexification stages. Post-convexification is based on the following result.

RESULT 5.1. *If J_w is a second-order-consistent working-set matrix associated with a symmetric H, then there exists a nonnegative $\bar{\sigma}$ such that the matrix $\bar{H} = H + \bar{\sigma} J_w^T J_w$ is positive definite. In addition, the solutions of the systems*

$$\begin{pmatrix} H & J_w^T \\ J_w & 0 \end{pmatrix} \begin{pmatrix} p \\ -\widehat{y}_w \end{pmatrix} = - \begin{pmatrix} g \\ c_w \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} \bar{H} & J_w^T \\ J_w & 0 \end{pmatrix} \begin{pmatrix} \bar{p} \\ -\bar{y}_w \end{pmatrix} = - \begin{pmatrix} g \\ c_w \end{pmatrix}$$

are related by the identities $\bar{p} = p$ and $\bar{y}_w = \widehat{y}_w - \bar{\sigma} c_w$. ∎

If the solution $(\widehat{x}_k, \widehat{y}_k)$ of the QP subproblem does not satisfy the descent condition, then $p_k = \widehat{x}_k - x_k$ is such that

$$p_k^T H(x_k, y_k) p_k = -g_L(x_k, \widehat{y}_k)^T p_k < \bar{\gamma} p_k^T p_k$$

for some positive $\bar{\gamma}$. The result implies that multipliers $\bar{y}_k$ such that $[\bar{y}_k]_i = 0$, for $i \notin \mathcal{W}$, and $[\bar{y}_k]_w = \widehat{y}_w - \bar{\sigma} c_w(x_k)$, provide the required curvature

$$p_k^T \bar{H}(x_k, y_k) p_k = -g_L(x_k, \bar{y}_k)^T p_k = \gamma p_k^T p_k,$$

where $\bar{\sigma} = \big(\gamma p_k^T p_k - p_k^T H(x_k, y_k) p_k\big) / \|c_w(x_k)\|^2$ with γ chosen such that $\gamma \geq \bar{\gamma}$. The extension of this result to the situation where $(\widehat{x}_k, \widehat{y}_k)$ satisfies the modified KKT system (5.5) is obvious.

6. Summary. The numerical results presented in Section 3 indicate that the optimization packages `SNOPT7` and `IPOPT` are very efficient and robust in general, solving over 85% of problems in the `CUTEst` test collection. However, the results also show that the performance of these codes depends greatly on the characteristics of the problem. These characteristics include the size of the problem, the availability of first and second derivatives, the types of constraints, and the availability of a good initial starting point. Ultimately, for every problem that is best solved by an SQP code, there will likely exist another that is best solved by an interior-point code.

To extend SQP methods so that second derivatives may be exploited reliably and efficiently, we propose convexification algorithms for the QP subproblem in an active-set SQP method for nonlinearly constrained optimization. Three forms of convexification are defined: preconvexification, concurrent convexification, and post-convexification. The methods require only minor changes to the algorithms used to solve the QP subproblem, and are designed so that modifications to the original problem are minimized and applied only when necessary.

It should be noted that the post-convexification Result 5.1 holds even if a conventional general QP method is used to solve the QP subproblem (provided that the method gives a final working set that is second-order consistent). It follows that post-convexification will define a descent direction regardless of whether concurrent convexification is used or not. The purpose of concurrent convexification is to reduce the probability of needing post-convexification, and to avoid the difficulties associated with solving an indefinite QP problem.

The methods defined here are the basis of the second-derivative solvers in the dense SQP package `DNOPT` of Gill, Saunders and Wong [18] and the forthcoming `SNOPT9`. All of the methods may be extended to problems in which the constraints are written in the form (3.1) (see Gill and Wong [20, Section 4]). In this case, the inequality constraints for the QP subproblem are upper and lower bounds, and all the modification matrices are diagonal.

Acknowledgments. We would like to thank Nick Gould for providing the latest version of the `CUTEst` test collection.

REFERENCES

[1] R. Byrd, J. Nocedal, R. Waltz, and Y. Wu, *On the use of piecewise linear models in nonlinear programming*, Math. Program., 137 (2013), pp. 289–324.

[2] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, *A limited memory algorithm for bound constrained optimization*, SIAM J. Sci. Comput., 16 (1995), pp. 1190–1208.

[3] L. B. Contesse, *Une caractérisation complète des minima locaux en programmation quadratique*, Numer. Math., 34 (1980), pp. 315–332.

[4] E. D. Dolan and J. J. Moré, *Benchmarking optimization software with COPS*, Technical Memorandum ANL/MCS-TM-246, Argonne National Laboratory, Argonne, IL, 2000.

[5] R. Fletcher, *An ℓ_1 penalty method for nonlinear constraints*, in Numerical Optimization 1984, P. T. Boggs, R. H. Byrd, and R. B. Schnabel, eds., Philadelphia, 1985, SIAM, pp. 26–40.

[6] R. Fletcher and S. Leyffer, *User manual for filterSQP*, Tech. Rep. NA/181, Dept. of Mathematics, University of Dundee, Scotland, 1998.

[7] A. Forsgren, *Inertia-controlling factorizations for optimization algorithms*, Appl. Num. Math., 43 (2002), pp. 91–107.

[8] A. Forsgren and P. E. Gill, *Primal-dual interior methods for nonconvex nonlinear programming*, SIAM J. Optim., 8 (1998), pp. 1132–1152.

[9] A. Forsgren, P. E. Gill, and W. Murray, *On the identification of local minimizers in inertia-controlling methods for quadratic programming*, SIAM J. Matrix Anal. Appl., 12 (1991), pp. 730–746.

[10] A. Forsgren and W. Murray, *Newton methods for large-scale linear equality-constrained minimization*, SIAM J. Matrix Anal. Appl., 14 (1993), pp. 560–587.

[11] P. E. Gill and M. W. Leonard, *Limited-memory reduced-Hessian methods for large-scale unconstrained optimization*, SIAM J. Optim., 14 (2003), pp. 380–401.

[12] P. E. Gill and W. Murray, *Newton-type methods for unconstrained and linearly constrained optimization*, Math. Program., 7 (1974), pp. 311–350.

[13] P. E. Gill, W. Murray, and M. A. Saunders, *SNOPT: An SQP algorithm for large-scale constrained optimization*, SIAM Rev., 47 (2005), pp. 99–131.

[14] ———, *User's guide for SQOPT Version 7: Software for large-scale linear and quadratic programming*, Numerical Analysis Report 06-1, Department of Mathematics, University of California, San Diego, La Jolla, CA, 2006.

[15] P. E. Gill, W. Murray, M. A. Saunders, and M. H. Wright, *User's guide for NPSOL (Version 4.0): a Fortran package for nonlinear programming*, Report SOL 86-2, Department of Operations Research, Stanford University, Stanford, CA, 1986.

[16] ———, *Some theoretical properties of an augmented Lagrangian merit function*, in Advances in Optimization and Parallel Computing, P. M. Pardalos, ed., North Holland, North Holland, 1992, pp. 101–128.

[17] P. E. Gill and D. P. Robinson, *A globally convergent stabilized SQP method*, SIAM J. Optim., 23 (2013), pp. 1983–2010.

[18] P. E. Gill, M. A. Saunders, and E. Wong, *User's Guide for DNOPT: a Fortran package for medium-scale nonlinear programming*, Center for Computational Mathematics Report CCoM 14-05, Department of Mathematics, University of California, San Diego, La Jolla, CA, 2014.

[19] P. E. Gill and E. Wong, *Sequential quadratic programming methods*, in Mixed Integer Nonlinear Programming, J. Lee and S. Leyffer, eds., vol. 154 of The IMA Volumes in Mathematics and its Applications, Springer New York, 2012, pp. 147–224. 10.1007/978-1-4614-1927-3_6.

[20] P. E. Gill and E. Wong, *Methods for convex and general quadratic programming*, Center for Computational Mathematics Report CCoM 14-03, University of California, San Diego, La Jolla, CA, 2014.

[21] P. E. Gill and E. Wong, *User's guide for SQIC: Software for large-scale quadratic programming*, Center for Computational Mathematics Report CCoM 14-02, Center for Computational Mathematics, University of California, San Diego, La Jolla, CA, 2014.

[22] N. I. M. Gould, *On modified factorizations for large-scale linearly constrained optimization*, SIAM J. Optim., 9 (1999), pp. 1041–1063.

[23] N. I. M. Gould, D. Orban, and P. L. Toint, *CUTEst: a constrained and unconstrained testing environment with safe threads*, technical report, Rutherford Appleton Laboratory, Chilton, England, 2013.

[24] N. I. M. Gould and D. P. Robinson, *A second derivative SQP method with imposed descent*, Numerical Analysis Report 08/09, Computational Laboratory, University of Oxford, Oxford, UK, 2008.
[25] ———, *A second derivative SQP method: Global convergence*, SIAM J. Optim., 20 (2010), pp. 2023–2048.
[26] ———, *A second derivative SQP method: Local convergence and practical issues*, SIAM J. Optim., 20 (2010), pp. 2049–2079.
[27] J. Greenstadt, *On the relative efficiencies of gradient methods*, Math. Comp., 21 (1967), pp. 360–367.
[28] L. Grippo, F. Lampariello, and S. Lucidi, *Newton-type algorithms with nonmonotone line search for large-scale unconstrained optimization*, in System modelling and optimization (Tokyo, 1987), vol. 113 of Lecture Notes in Control and Inform. Sci., Springer, Berlin, 1988, pp. 187–196.
[29] ———, *A truncated Newton method with nonmonotone line search for unconstrained optimization*, J. Optim. Theory Appl., 60 (1989), pp. 401–419.
[30] ———, *A class of nonmonotone stabilization methods in unconstrained optimization*, Numer. Math., 59 (1991), pp. 779–805.
[31] S. P. Han, *A globally convergent method for nonlinear programming*, J. Optim. Theory Appl., 22 (1977), pp. 297–309.
[32] V. Kungurtsev, *Second-Derivative Sequential Quadratic Programming Methods for Nonlinear Optimization*, PhD thesis, Department of Mathematics, University of California San Diego, La Jolla, CA, 2013.
[33] C. C. Paige and M. A. Saunders, *Solution of sparse indefinite systems of linear equations*, SIAM J. Numer. Anal., 12 (1975), pp. 617–629.
[34] M. J. D. Powell, *A fast algorithm for nonlinearly constrained optimization calculations*, in Numerical Analysis, Dundee 1977, G. A. Watson, ed., no. 630 in Lecture Notes in Mathematics, Heidelberg, Berlin, New York, 1978, Springer Verlag, pp. 144–157.
[35] K. Schittkowski, *The nonlinear programming method of Wilson, Han, and Powell with an augmented Lagrangian type line search function. I. Convergence analysis*, Numer. Math., 38 (1981/82), pp. 83–114.
[36] K. Schittkowski, *NLPQL: a Fortran subroutine for solving constrained nonlinear programming problems*, Ann. Oper. Res., 11 (1985/1986), pp. 485–500.
[37] R. B. Schnabel and E. Eskow, *A new modified Cholesky factorization*, SIAM J. Sci. and Statist. Comput., 11 (1990), pp. 1136–1158.
[38] P. Spellucci, *An SQP method for general nonlinear programs using only equality constrained subproblems*, Math. Program., 82 (1998), pp. 413–448.
[39] P. L. Toint, *An assessment of nonmonotone linesearch techniques for unconstrained optimization*, SIAM J. Sci. Comput., 17 (1996), pp. 725–739.
[40] A. Wächter and L. T. Biegler, *Line search filter methods for nonlinear programming: local convergence*, SIAM J. Optim., 16 (2005), pp. 32–48 (electronic).
[41] ———, *Line search filter methods for nonlinear programming: motivation and global convergence*, SIAM J. Optim., 16 (2005), pp. 1–31 (electronic).
[42] ———, *On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming*, Math. Program., 106 (2006), pp. 25–57.
[43] A. Wächter, L. T. Biegler, Y.-D. Lang, and A. Raghunathan, *IPOPT: An interior point algorithm for large-scale nonlinear optimization.* https://projects.coin-or.org/Ipopt, 2002.
[44] H. Zhang and W. W. Hager, *A nonmonotone line search technique and its application to unconstrained optimization*, SIAM J. Optim., 14 (2004), pp. 1043–1056 (electronic).