

CHAPTER *18*

Sequential Quadratic Programming

One of the most effective methods for nonlinearly constrained optimization generates steps by solving quadratic subproblems. This sequential quadratic programming (SQP) approach can be used both in line search and trust-region frameworks, and is appropriate for small or large problems. Unlike linearly constrained Lagrangian methods (Chapter 17), which are effective when most of the constraints are linear, SQP methods show their strength when solving problems with significant nonlinearities in the constraints.

All the methods considered in this chapter are active-set methods; a more descriptive title for this chapter would perhaps be “Active-Set Methods for Nonlinear Programming.”

In Chapter 14 we study interior-point methods for nonlinear programming, a competing approach for handling inequality-constrained problems.

There are two types of active-set SQP methods. In the IQP approach, a general inequality-constrained quadratic program is solved at each iteration, with the twin goals of computing a step and generating an estimate of the optimal active set. EQP methods decouple these computations. They first compute an estimate of the optimal active set, then solve an equality-constrained quadratic program to find the step. In this chapter we study both IQP and EQP methods.

Our development of SQP methods proceeds in two stages. First, we consider local methods that motivate the SQP approach and allow us to introduce the step computation techniques in a simple setting. Second, we consider practical line search and trust-region methods that achieve convergence from remote starting points. Throughout the chapter we give consideration to the algorithmic demands of solving large problems.

18.1 LOCAL SQP METHOD

We begin by considering the equality-constrained problem

$$\min f(x) \tag{18.1a}$$

$$\text{subject to } c(x) = 0, \tag{18.1b}$$

where $f : \mathbb{R}^n \rightarrow \mathbb{R}$ and $c : \mathbb{R}^n \rightarrow \mathbb{R}^m$ are smooth functions. The idea behind the SQP approach is to model (18.1) at the current iterate x_k by a quadratic programming subproblem, then use the minimizer of this subproblem to define a new iterate x_{k+1} . The challenge is to design the quadratic subproblem so that it yields a good step for the nonlinear optimization problem. Perhaps the simplest derivation of SQP methods, which we present now, views them as an application of Newton's method to the KKT optimality conditions for (18.1).

From (12.33), we know that the Lagrangian function for this problem is $\mathcal{L}(x, \lambda) = f(x) - \lambda^T c(x)$. We use $A(x)$ to denote the Jacobian matrix of the constraints, that is,

$$A(x)^T = [\nabla c_1(x), \nabla c_2(x), \dots, \nabla c_m(x)], \tag{18.2}$$

where $c_i(x)$ is the i th component of the vector $c(x)$. The first-order (KKT) conditions (12.34) of the equality-constrained problem (18.1) can be written as a system of $n + m$ equations in the $n + m$ unknowns x and λ :

$$F(x, \lambda) = \begin{bmatrix} \nabla f(x) - A(x)^T \lambda \\ c(x) \end{bmatrix} = 0. \tag{18.3}$$

Any solution (x^*, λ^*) of the equality-constrained problem (18.1) for which $A(x^*)$ has full

rank satisfies (18.3). One approach that suggests itself is to solve the nonlinear equations (18.3) by using Newton's method, as described in Chapter 11.

The Jacobian of (18.3) with respect to x and λ is given by

$$F'(x, \lambda) = \begin{bmatrix} \nabla_{xx}^2 \mathcal{L}(x, \lambda) & -A(x)^T \\ A(x) & 0 \end{bmatrix}. \quad (18.4)$$

The Newton step from the iterate (x_k, λ_k) is thus given by

$$\begin{bmatrix} x_{k+1} \\ \lambda_{k+1} \end{bmatrix} = \begin{bmatrix} x_k \\ \lambda_k \end{bmatrix} + \begin{bmatrix} p_k \\ p_\lambda \end{bmatrix}, \quad (18.5)$$

where p_k and p_λ solve the Newton–KKT system

$$\begin{bmatrix} \nabla_{xx}^2 \mathcal{L}_k & -A_k^T \\ A_k & 0 \end{bmatrix} \begin{bmatrix} p_k \\ p_\lambda \end{bmatrix} = \begin{bmatrix} -\nabla f_k + A_k^T \lambda_k \\ -c_k \end{bmatrix}. \quad (18.6)$$

This Newton iteration is well defined when the KKT matrix in (18.6) is nonsingular. We saw in Chapter 16 that this matrix is nonsingular if the following assumption holds at $(x, \lambda) = (x_k, \lambda_k)$.

Assumptions 18.1.

- (a) The constraint Jacobian $A(x)$ has full row rank;
- (b) The matrix $\nabla_{xx}^2 \mathcal{L}(x, \lambda)$ is positive definite on the tangent space of the constraints, that is, $d^T \nabla_{xx}^2 \mathcal{L}(x, \lambda) d > 0$ for all $d \neq 0$ such that $A(x)d = 0$.

The first assumption is the linear independence constraint qualification discussed in Chapter 12 (see Definition 12.4), which we assume throughout this chapter. The second condition holds whenever (x, λ) is close to the optimum (x^*, λ^*) and the second-order sufficient condition is satisfied at the solution (see Theorem 12.6). The Newton iteration (18.5), (18.6) can be shown to be quadratically convergent under these assumptions (see Theorem 18.4) and constitutes an excellent algorithm for solving equality-constrained problems, provided that the starting point is close enough to x^* .

SQP FRAMEWORK

There is an alternative way to view the iteration (18.5), (18.6). Suppose that at the iterate (x_k, λ_k) we model problem (18.1) using the quadratic program

$$\min_p \quad f_k + \nabla f_k^T p + \frac{1}{2} p^T \nabla_{xx}^2 \mathcal{L}_k p \quad (18.7a)$$

$$\text{subject to} \quad A_k p + c_k = 0. \quad (18.7b)$$

If Assumptions 18.1 hold, this problem has a unique solution (p_k, l_k) that satisfies

$$\nabla_{xx}^2 \mathcal{L}_k p_k + \nabla f_k - A_k^T l_k = 0, \quad (18.8a)$$

$$A_k p_k + c_k = 0. \quad (18.8b)$$

The vectors p_k and l_k can be identified with the solution of the Newton equations (18.6). If we subtract $A_k^T \lambda_k$ from both sides of the first equation in (18.6), we obtain

$$\begin{bmatrix} \nabla_{xx}^2 \mathcal{L}_k & -A_k^T \\ A_k & 0 \end{bmatrix} \begin{bmatrix} p_k \\ \lambda_{k+1} \end{bmatrix} = \begin{bmatrix} -\nabla f_k \\ -c_k \end{bmatrix}. \quad (18.9)$$

Hence, by nonsingularity of the coefficient matrix, we have that $\lambda_{k+1} = l_k$ and that p_k solves (18.7) and (18.6).

The new iterate (x_{k+1}, λ_{k+1}) can therefore be defined either as the solution of the quadratic program (18.7) or as the iterate generated by Newton's method (18.5), (18.6) applied to the optimality conditions of the problem. Both viewpoints are useful. The Newton point of view facilitates the analysis, whereas the SQP framework enables us to derive practical algorithms and to extend the technique to the inequality-constrained case.

We now state the SQP method in its simplest form.

Algorithm 18.1 (Local SQP Algorithm for solving (18.1)).

Choose an initial pair (x_0, λ_0) ; set $k \leftarrow 0$;
repeat until a convergence test is satisfied
 Evaluate $f_k, \nabla f_k, \nabla_{xx}^2 \mathcal{L}_k, c_k$, and A_k ;
 Solve (18.7) to obtain p_k and l_k ;
 Set $x_{k+1} \leftarrow x_k + p_k$ and $\lambda_{k+1} \leftarrow l_k$;
end (repeat)

We note in passing that, in the objective (18.7a) of the quadratic program, we could replace the linear term $\nabla f_k^T p$ by $\nabla_x \mathcal{L}(x_k, \lambda_k)^T p$, since the constraint (18.7b) makes the two choices equivalent. In this case, (18.7a) is a quadratic approximation of the Lagrangian function. This fact provides a motivation for our choice of the quadratic model (18.7): We first replace the nonlinear program (18.1) by the problem of minimizing the Lagrangian subject to the equality constraints (18.1b), then make a quadratic approximation to the Lagrangian and a linear approximation to the constraints to obtain (18.7).

INEQUALITY CONSTRAINTS

The SQP framework can be extended easily to the general nonlinear programming problem

$$\min f(x) \quad (18.10a)$$

$$\text{subject to } c_i(x) = 0, \quad i \in \mathcal{E}, \quad (18.10b)$$

$$c_i(x) \geq 0, \quad i \in \mathcal{I}. \quad (18.10c)$$

To model this problem we now linearize both the inequality and equality constraints to obtain

$$\min_p \quad f_k + \nabla f_k^T p + \frac{1}{2} p^T \nabla_{xx}^2 \mathcal{L}_k p \quad (18.11a)$$

$$\text{subject to} \quad \nabla c_i(x_k)^T p + c_i(x_k) = 0, \quad i \in \mathcal{E}, \quad (18.11b)$$

$$\nabla c_i(x_k)^T p + c_i(x_k) \geq 0, \quad i \in \mathcal{I}. \quad (18.11c)$$

We can use one of the algorithms for quadratic programming described in Chapter 16 to solve this problem. The new iterate is given by $(x_k + p_k, \lambda_{k+1})$ where p_k and λ_{k+1} are the solution and the corresponding Lagrange multiplier of (18.11). A local SQP method for (18.10) is thus given by Algorithm 18.1 with the modification that the step is computed from (18.11).

In this IQP approach the set of active constraints \mathcal{A}_k at the solution of (18.11) constitutes our guess of the active set at the solution of the nonlinear program. If the SQP method is able to correctly identify this optimal active set (and not change its guess at a subsequent iteration) then it will act like a Newton method for equality-constrained optimization and will converge rapidly. The following result gives conditions under which this desirable behavior takes place. Recall that strict complementarity is said to hold at a solution pair (x^*, λ^*) if there is no index $i \in \mathcal{I}$ such that $\lambda_i^* = c_i(x^*) = 0$.

Theorem 18.1 (Robinson [267]).

Suppose that x^ is a local solution of (18.10) at which the KKT conditions are satisfied for some λ^* . Suppose, too, that the linear independence constraint qualification (LICQ) (Definition 12.4), the strict complementarity condition (Definition 12.5), and the second-order sufficient conditions (Theorem 12.6) hold at (x^*, λ^*) . Then if (x_k, λ_k) is sufficiently close to (x^*, λ^*) , there is a local solution of the subproblem (18.11) whose active set \mathcal{A}_k is the same as the active set $\mathcal{A}(x^*)$ of the nonlinear program (18.10) at x^* .*

It is also remarkable that, far from the solution, the SQP approach is usually able to improve the estimate of the active set and guide the iterates toward a solution; see Section 18.7.

18.2 PREVIEW OF PRACTICAL SQP METHODS

IQP AND EQP

There are two ways of designing SQP methods for solving the general nonlinear programming problem (18.10). The first is the approach just described, which solves at

every iteration the quadratic subprogram (18.11), taking the active set at the solution of this subproblem as a guess of the optimal active set. This approach is referred to as the IQP (inequality-constrained QP) approach; it has proved to be quite successful in practice. Its main drawback is the expense of solving the general quadratic program (18.11), which can be high when the problem is large. As the iterates of the SQP method converge to the solution, however, solving the quadratic subproblem becomes economical if we use information from the previous iteration to make a good guess of the optimal solution of the current subproblem. This *warm-start* strategy is described below.

The second approach selects a subset of constraints at each iteration to be the so-called working set, and solves only equality-constrained subproblems of the form (18.7), where the constraints in the working sets are imposed as equalities and all other constraints are ignored. The working set is updated at every iteration by rules based on Lagrange multiplier estimates, or by solving an auxiliary subproblem. This EQP (equality-constrained QP) approach has the advantage that the equality-constrained quadratic subproblems are less expensive to solve than (18.11) in the large-scale case.

An example of an EQP method is the sequential linear-quadratic programming (SLQP) method discussed in Section 18.5. This approach constructs a linear program by omitting the quadratic term $p^T \nabla_{xx}^2 \mathcal{L}_k p$ from (18.11a) and adding a trust-region constraint $\|p\|_\infty \leq \Delta_k$ to the subproblem. The active set of the resulting linear programming subproblem is taken to be the working set for the current iteration. The method then fixes the constraints in the working set and solves an equality-constrained quadratic program (with the term $p^T \nabla_{xx}^2 \mathcal{L}_k p$ reinserted) to obtain the SQP step. Another successful EQP method is the gradient projection method described in Section 16.7 in the context of bound constrained quadratic programs. In this method, the working set is determined by minimizing a quadratic model along the path obtained by projecting the steepest descent direction onto the feasible region.

ENFORCING CONVERGENCE

To be practical, an SQP method must be able to converge from remote starting points and on nonconvex problems. We now outline how the local SQP strategy can be adapted to meet these goals.

We begin by drawing an analogy with unconstrained optimization. In its simplest form, the Newton iteration for minimizing a function f takes a step to the minimizer of the quadratic model

$$m_k(p) = f_k + \nabla f_k^T p + \frac{1}{2} p^T \nabla^2 f_k p.$$

This framework is useful near the solution, where the Hessian $\nabla^2 f(x_k)$ is normally positive definite and the quadratic model has a well defined minimizer. When x_k is not close to the solution, however, the model function m_k may not be convex. Trust-region methods ensure that the new iterate is always well defined and useful by restricting the candidate step p_k

to some neighborhood of the origin. Line search methods modify the Hessian in $m_k(p)$ to make it positive definite (possibly replacing it by a quasi-Newton approximation B_k), to ensure that p_k is a descent direction for the objective function f .

Similar strategies are used to globalize SQP methods. If $\nabla_{xx}^2 \mathcal{L}_k$ is positive definite on the tangent space of the active constraints, the quadratic subproblem (18.7) has a unique solution. When $\nabla_{xx}^2 \mathcal{L}_k$ does not have this property, line search methods either replace it by a positive definite approximation B_k or modify $\nabla_{xx}^2 \mathcal{L}_k$ directly during the process of matrix factorization. In all these cases, the subproblem (18.7) becomes well defined, but the modifications may introduce unwanted distortions in the model.

Trust-region SQP methods add a constraint to the subproblem, limiting the step to a region within which the model (18.7) is considered reliable. These methods are able to handle indefinite Hessians $\nabla_{xx}^2 \mathcal{L}_k$. The inclusion of the trust region may, however, cause the subproblem to become infeasible, and the procedures for handling this situation complicate the algorithms and increase their computational cost. Due to these tradeoffs, neither of the two SQP approaches—line search or trust-region—is currently regarded as clearly superior to the other.

The technique used to accept or reject steps also impacts the efficiency of SQP methods. In unconstrained optimization, the merit function is simply the objective f , and it remains fixed throughout the minimization procedure. For constrained problems, we use devices such as a merit function or a filter (see Section 15.4). The parameters or entries used in these devices must be updated in a way that is compatible with the step produced by the SQP method.

18.3 ALGORITHMIC DEVELOPMENT

In this section we expand on the ideas of the previous section and describe various ingredients needed to produce practical SQP algorithms. We focus on techniques for ensuring that the subproblems are always feasible, on alternative choices for the Hessian of the quadratic model, and on step-acceptance mechanisms.

HANDLING INCONSISTENT LINEARIZATIONS

A possible difficulty with SQP methods is that the linearizations (18.11b), (18.11c) of the nonlinear constraints may give rise to an infeasible subproblem. Consider, for example, the case in which $n = 1$ and the constraints are $x \leq 1$ and $x^2 \geq 4$. When we linearize these constraints at $x_k = 1$, we obtain the inequalities

$$-p \geq 0 \quad \text{and} \quad 2p - 3 \geq 0,$$

which are inconsistent.

To overcome this difficulty, we can reformulate the nonlinear program (18.10) as the ℓ_1 penalty problem

$$\min_{x, v, w, t} \quad f(x) + \mu \sum_{i \in \mathcal{E}} (v_i + w_i) + \mu \sum_{i \in \mathcal{I}} t_i \quad (18.12a)$$

$$\text{subject to} \quad c_i(x) = v_i - w_i, \quad i \in \mathcal{E}, \quad (18.12b)$$

$$c_i(x) \geq -t_i, \quad i \in \mathcal{I}, \quad (18.12c)$$

$$v, w, t \geq 0, \quad (18.12d)$$

for some positive choice of the penalty parameter μ . The quadratic subproblem (18.11) associated with (18.12) is always feasible. As discussed in Chapter 17, if the nonlinear problem (18.10) has a solution x^* that satisfies certain regularity assumptions, and if the penalty parameter μ is sufficiently large, then x^* (along with $v_i^* = w_i^* = 0, i \in \mathcal{E}$ and $t_i^* = 0, i \in \mathcal{I}$) is a solution of the penalty problem (18.12). If, on the other hand, there is no feasible solution to the nonlinear problem and μ is large enough, then the penalty problem (18.12) usually determines a stationary point of the infeasibility measure. The choice of μ has been discussed in Chapter 17 and is considered again in Section 18.5. The SNOPT software package [127] uses the formulation (18.12), which is sometimes called the *elastic mode*, to deal with inconsistencies of the linearized constraints.

Other procedures for relaxing the constraints are presented in Section 18.5 in the context of trust-region methods.

FULL QUASI-NEWTON APPROXIMATIONS

The Hessian of the Lagrangian $\nabla_{xx}^2 \mathcal{L}(x_k, \lambda_k)$ is made up of second derivatives of the objective function and constraints. In some applications, this information is not easy to compute, so it is useful to consider replacing the Hessian $\nabla_{xx}^2 \mathcal{L}(x_k, \lambda_k)$ in (18.11a) by a quasi-Newton approximation. Since the BFGS and SR1 formulae have proved to be successful in the context of unconstrained optimization, we can employ them here as well.

The update for B_k that results from the step from iterate k to iterate $k + 1$ makes use of the vectors s_k and y_k defined as follows:

$$s_k = x_{k+1} - x_k, \quad y_k = \nabla_x \mathcal{L}(x_{k+1}, \lambda_{k+1}) - \nabla_x \mathcal{L}(x_k, \lambda_{k+1}). \quad (18.13)$$

We compute the new approximation B_{k+1} using the BFGS or SR1 formulae given, respectively, by (6.19) and (6.24). We can view this process as the application of quasi-Newton updating to the case in which the objective function is given by the Lagrangian $\mathcal{L}(x, \lambda)$ (with λ fixed). This viewpoint immediately reveals the strengths and weaknesses of this approach.

If $\nabla_{xx}^2 \mathcal{L}$ is positive definite in the region where the minimization takes place, then BFGS quasi-Newton approximations B_k will reflect some of the curvature information of the problem, and the iteration will converge robustly and rapidly, just as in the unconstrained BFGS method. If, however, $\nabla_{xx}^2 \mathcal{L}$ contains negative eigenvalues, then the BFGS approach

of approximating it with a positive definite matrix may be problematic. BFGS updating requires that s_k and y_k satisfy the curvature condition $s_k^T y_k > 0$, which may not hold when s_k and y_k are defined by (18.13), even when the iterates are close to the solution.

To overcome this difficulty, we could *skip* the BFGS update if the condition

$$s_k^T y_k \geq \theta s_k^T B_k s_k \quad (18.14)$$

is not satisfied, where θ is a positive parameter (10^{-2} , say). This strategy may, on occasion, yield poor performance or even failure, so it cannot be regarded as adequate for general-purpose algorithms.

A more effective modification ensures that the update is always well defined by modifying the definition of y_k .

Procedure 18.2 (Damped BFGS Updating).

Given: symmetric and positive definite matrix B_k ;

Define s_k and y_k as in (18.13) and set

$$r_k = \theta_k y_k + (1 - \theta_k) B_k s_k,$$

where the scalar θ_k is defined as

$$\theta_k = \begin{cases} 1 & \text{if } s_k^T y_k \geq 0.2 s_k^T B_k s_k, \\ (0.8 s_k^T B_k s_k) / (s_k^T B_k s_k - s_k^T y_k) & \text{if } s_k^T y_k < 0.2 s_k^T B_k s_k; \end{cases} \quad (18.15)$$

Update B_k as follows:

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{r_k r_k^T}{s_k^T r_k}. \quad (18.16)$$

The formula (18.16) is simply the standard BFGS update formula, with y_k replaced by r_k . It guarantees that B_{k+1} is positive definite, since it is easy to show that when $\theta_k \neq 1$ we have

$$s_k^T r_k = 0.2 s_k^T B_k s_k > 0. \quad (18.17)$$

To gain more insight into this strategy, note that the choice $\theta_k = 0$ gives $B_{k+1} = B_k$, while $\theta_k = 1$ gives the (possibly indefinite) matrix produced by the unmodified BFGS update. A value $\theta_k \in (0, 1)$ thus produces a matrix that interpolates the current approximation B_k and the one produced by the unmodified BFGS formula. The choice of θ_k ensures that the new approximation stays close enough to the current approximation B_k to ensure positive definiteness.

Damped BFGS updating often works well but it, too, can behave poorly on difficult problems. It still fails to address the underlying problem that the Lagrangian Hessian may not be positive definite. For this reason, SR1 updating may be more appropriate, and is indeed a good choice for trust-region SQP methods. An SR1 approximation to the Hessian of the Lagrangian is obtained by applying formula (6.24) with s_k and y_k defined by (18.13), using the safeguards described in Chapter 6. Line search methods cannot, however, accept indefinite Hessian approximations and would therefore need to modify the SR1 formula, possibly by adding a sufficiently large multiple of the identity matrix; see the discussion around (19.25).

All quasi-Newton approximations B_k discussed above are dense $n \times n$ matrices that can be expensive to store and manipulate in the large-scale case. Limited-memory updating is useful in this context and is often implemented in software packages. (See (19.29) for an implementation of limited-memory BFGS in a constrained optimization algorithm.)

REDUCED-HESSIAN QUASI-NEWTON APPROXIMATIONS

When we examine the KKT system (18.9) for the equality-constrained problem (18.1), we see that the part of the step p_k in the range space of A_k^T is completely determined by the second block row $A_k p_k = -c_k$. The Lagrangian Hessian $\nabla_{xx}^2 \mathcal{L}_k$ affects only the part of p_k in the orthogonal subspace, namely, the null space of A_k . It is reasonable, therefore, to consider quasi-Newton methods that find approximations to only that part of $\nabla_{xx}^2 \mathcal{L}_k$ that affects the component of p_k in the null space of A_k . In this section, we consider quasi-Newton methods based on these reduced-Hessian approximations. Our focus is on equality-constrained problems in this section, as existing SQP methods for the full problem (18.10) use reduced-Hessian approaches only after an equality-constrained subproblem has been generated.

To derive reduced-Hessian methods, we consider solution of the step equations (18.9) by means of the null space approach of Section 16.2. In that section, we defined matrices Y_k and Z_k whose columns span the range space of A_k^T and the null space of A_k , respectively. By writing

$$p_k = Y_k p_Y + Z_k p_Z, \quad (18.18)$$

and substituting into (18.9), we obtain the following system to be solved for p_Y and p_Z :

$$(A_k Y_k) p_Y = -c_k, \quad (18.19a)$$

$$(Z_k^T \nabla_{xx}^2 \mathcal{L}_k Z_k) p_Z = -Z_k^T \nabla_{xx}^2 \mathcal{L}_k Y_k p_Y - Z_k^T \nabla f_k. \quad (18.19b)$$

From the first block of equations in (18.9) we see that the Lagrange multipliers λ_{k+1} , which are sometimes called QP multipliers, can be obtained by solving

$$(A_k Y_k)^T \lambda_{k+1} = Y_k^T (\nabla f_k + \nabla_{xx}^2 \mathcal{L}_k p_k). \quad (18.20)$$

We can avoid computation of the Hessian $\nabla_{xx}^2 \mathcal{L}_k$ by introducing several approximations in the null-space approach. First, we delete the term involving p_k from the right-hand-side of (18.20), thereby decoupling the computations of p_k and λ_{k+1} and eliminating the need for $\nabla_{xx}^2 \mathcal{L}_k$ in this term. This simplification can be justified by observing that p_k converges to zero as we approach the solution, whereas ∇f_k normally does not. Therefore, the multipliers computed in this manner will be good estimates of the QP multipliers near the solution. More specifically, if we choose $Y_k = A_k^T$ (which is a valid choice for Y_k when A_k has full row rank; see (15.16)), we obtain

$$\hat{\lambda}_{k+1} = (A_k A_k^T)^{-1} A_k \nabla f_k. \quad (18.21)$$

These are called the *least-squares multipliers* because they can also be derived by solving the problem

$$\min_{\lambda} \|\nabla_x \mathcal{L}(x_k, \lambda)\|_2^2 = \|\nabla f_k - A_k^T \lambda\|_2^2. \quad (18.22)$$

This observation shows that the least-squares multipliers are useful even when the current iterate is far from the solution, because they seek to satisfy the first-order optimality condition in (18.3) as closely as possible. Conceptually, the use of least-squares multipliers transforms the SQP method from a primal-dual iteration in x and λ to a purely primal iteration in the x variable alone.

Our second simplification of the null-space approach is to remove the cross term $Z_k^T \nabla_{xx}^2 \mathcal{L}_k Y_k p_Y$ in (18.19b), thereby yielding the simpler system

$$(Z_k^T \nabla_{xx}^2 \mathcal{L}_k Z_k) p_Z = -Z_k^T \nabla f_k. \quad (18.23)$$

This approach has the advantage that it needs to approximate only the matrix $Z_k^T \nabla_{xx}^2 \mathcal{L}_k Z_k$, not the $(n - m) \times m$ cross-term matrix $Z_k^T \nabla_{xx}^2 \mathcal{L}_k Y_k$, which is a relatively large matrix when $m \gg n - m$. Dropping the cross term is justified when $Z_k^T \nabla_{xx}^2 \mathcal{L}_k Z_k$ is replaced by a quasi-Newton approximation because the normal component p_Y usually converges to zero faster than the tangential component p_Z , thereby making (18.23) a good approximation of (18.19b).

Having dispensed with the partial Hessian $Z_k^T \nabla_{xx}^2 \mathcal{L}_k Y_k$, we discuss how to approximate the remaining part $Z_k^T \nabla_{xx}^2 \mathcal{L}_k Z_k$. Suppose we have just taken a step $\alpha_k p_k = x_{k+1} - x_k = \alpha_k Z_k p_Z + \alpha_k Y_k p_Y$. By Taylor's theorem, writing $\nabla_{xx}^2 \mathcal{L}_{k+1} = \nabla_{xx}^2 \mathcal{L}(x_{k+1}, \lambda_{k+1})$, we have

$$\nabla_{xx}^2 \mathcal{L}_{k+1} \alpha_k p_k \approx \nabla_x \mathcal{L}(x_k + \alpha_k p_k, \lambda_{k+1}) - \nabla_x \mathcal{L}(x_k, \lambda_{k+1}).$$

By premultiplying by Z_k^T , we have

$$\begin{aligned} & Z_k^T \nabla_{xx}^2 \mathcal{L}_{k+1} Z_k \alpha_k p_Z \\ & \approx -Z_k^T \nabla_{xx}^2 \mathcal{L}_{k+1} Y_k \alpha_k p_Y + Z_k^T [\nabla_x \mathcal{L}(x_k + \alpha_k p_k, \lambda_{k+1}) - \nabla_x \mathcal{L}(x_k, \lambda_{k+1})]. \end{aligned} \quad (18.24)$$

If we drop the cross term $Z_k^T \nabla_{xx}^2 \mathcal{L}_{k+1} Y_k \alpha_k p_Y$ (using the rationale discussed earlier), we see that the secant equation for M_k can be defined by

$$M_{k+1} s_k = y_k, \quad (18.25)$$

where s_k and y_k are given by

$$s_k = \alpha_k p_Z, \quad y_k = Z_k^T [\nabla_x \mathcal{L}(x_k + \alpha_k p_k, \lambda_{k+1}) - \nabla_x \mathcal{L}(x_k, \lambda_{k+1})]. \quad (18.26)$$

We then apply the BFGS or SR1 formulae, using these definitions for the correction vectors s_k and y_k , to define the new approximation M_{k+1} . An advantage of this reduced-Hessian approach, compared to full-Hessian quasi-Newton approximations, is that the reduced Hessian is much more likely to be positive definite, even when the current iterate is some distance from the solution. When using the BFGS formula, the safeguarding mechanism discussed above will be required less often in line search implementations.

MERIT FUNCTIONS

SQP methods often use a merit function to decide whether a trial step should be accepted. In line search methods, the merit function controls the size of the step; in trust-region methods it determines whether the step is accepted or rejected and whether the trust-region radius should be adjusted. A variety of merit functions have been used in SQP methods, including nonsmooth penalty functions and augmented Lagrangians. We limit our discussion to exact, nonsmooth merit functions typified by the ℓ_1 merit function discussed in Chapters 15 and 17.

For the purpose of step computation and evaluation of a merit function, inequality constraints $c(x) \geq 0$ are often converted to the form

$$\bar{c}(x, s) = c(x) - s = 0,$$

where $s \geq 0$ is a vector of slacks. (The condition $s \geq 0$ is typically not monitored by the merit function.) Therefore, in the discussion that follows we assume that all constraints are in the form of equalities, and we focus our attention on problem (18.1).

The ℓ_1 merit function for (18.1) takes the form

$$\phi_1(x; \mu) = f(x) + \mu \|c(x)\|_1. \quad (18.27)$$

In a line search method, a step $\alpha_k p_k$ will be accepted if the following sufficient decrease condition holds:

$$\phi_1(x_k + \alpha_k p_k; \mu_k) \leq \phi_1(x_k, \mu_k) + \eta \alpha_k D(\phi_1(x_k; \mu); p_k), \quad \eta \in (0, 1), \quad (18.28)$$

where $D(\phi_1(x_k; \mu); p_k)$ denotes the directional derivative of ϕ_1 in the direction p_k . This requirement is analogous to the Armijo condition (3.4) for unconstrained optimization provided that p_k is a descent direction, that is, $D(\phi_1(x_k; \mu); p_k) < 0$. This descent condition holds if the penalty parameter μ is chosen sufficiently large, as we show in the following result.

Theorem 18.2.

Let p_k and λ_{k+1} be generated by the SQP iteration (18.9). Then the directional derivative of ϕ_1 in the direction p_k satisfies

$$D(\phi_1(x_k; \mu); p_k) = \nabla f_k^T p_k - \mu \|c_k\|_1. \quad (18.29)$$

Moreover, we have that

$$D(\phi_1(x_k; \mu); p_k) \leq -p_k^T \nabla_{xx}^2 \mathcal{L}_k p_k - (\mu - \|\lambda_{k+1}\|_\infty) \|c_k\|_1. \quad (18.30)$$

PROOF. By applying Taylor's theorem (see (2.5)) to f and c_i , $i = 1, 2, \dots, m$, we obtain

$$\begin{aligned} \phi_1(x_k + \alpha p; \mu) - \phi_1(x_k; \mu) &= f(x_k + \alpha p) - f_k + \mu \|c(x_k + \alpha p)\|_1 - \mu \|c_k\|_1 \\ &\leq \alpha \nabla f_k^T p + \gamma \alpha^2 \|p\|^2 + \mu \|c_k + \alpha A_k p\|_1 - \mu \|c_k\|_1, \end{aligned}$$

where the positive constant γ bounds the second-derivative terms in f and c . If $p = p_k$ is given by (18.9), we have that $A_k p_k = -c_k$, so for $\alpha \leq 1$ we have that

$$\phi_1(x_k + \alpha p_k; \mu) - \phi_1(x_k; \mu) \leq \alpha [\nabla f_k^T p_k - \mu \|c_k\|_1] + \alpha^2 \gamma \|p_k\|^2.$$

By arguing similarly, we also obtain the following lower bound:

$$\phi_1(x_k + \alpha p_k; \mu) - \phi_1(x_k; \mu) \geq \alpha [\nabla f_k^T p_k - \mu \|c_k\|_1] - \alpha^2 \gamma \|p_k\|^2.$$

Taking limits, we conclude that the directional derivative of ϕ_1 in the direction p_k is given by

$$D(\phi_1(x_k; \mu); p_k) = \nabla f_k^T p_k - \mu \|c_k\|_1, \quad (18.31)$$

which proves (18.29). The fact that p_k satisfies the first equation in (18.9) implies that

$$D(\phi_1(x_k; \mu); p_k) = -p_k^T \nabla_{xx}^2 \mathcal{L}_k p_k + p_k^T A_k^T \lambda_{k+1} - \mu \|c_k\|_1.$$

From the second equation in (18.9), we can replace the term $p_k^T A_k^T \lambda_{k+1}$ in this expression by $-c_k^T \lambda_{k+1}$. By making this substitution in the expression above and invoking the inequality

$$-c_k^T \lambda_{k+1} \leq \|c_k\|_1 \|\lambda_{k+1}\|_\infty,$$

we obtain (18.30). □

It follows from (18.30) that p_k will be a descent direction for ϕ_1 if $p_k \neq 0$, $\nabla_{xx}^2 \mathcal{L}_k$ is positive definite and

$$\mu > \|\lambda_{k+1}\|_\infty. \quad (18.32)$$

(A more detailed analysis shows that this assumption on $\nabla_{xx}^2 \mathcal{L}_k$ can be relaxed; we need only the reduced Hessian $Z_k^T \nabla_{xx}^2 \mathcal{L}_k Z_k$ to be positive definite.)

One strategy for choosing the new value of the penalty parameter μ in $\phi_1(x; \mu)$ at every iteration is to increase the previous value, if necessary, so as to satisfy (18.32), with some margin. It has been observed, however, that this strategy may select inappropriate values of μ and often interferes with the progress of the iteration.

An alternative approach, based on (18.29), is to require that the directional derivative be sufficiently negative in the sense that

$$D(\phi_1(x_k; \mu); p_k) = \nabla f_k^T p_k - \mu \|c_k\|_1 \leq -\rho \mu \|c_k\|_1,$$

for some $\rho \in (0, 1)$. This inequality holds if

$$\mu \geq \frac{\nabla f_k^T p_k}{(1 - \rho) \|c_k\|_1}. \quad (18.33)$$

This choice is not dependent on the Lagrange multipliers and performs adequately in practice.

A more effective strategy for choosing μ , which is appropriate both in the line search and trust-region contexts, considers the effect of the step on a model of the merit function. We define a (piecewise) quadratic model of ϕ_1 by

$$q_\mu(p) = f_k + \nabla f_k^T p + \frac{\sigma}{2} p^T \nabla_{xx}^2 \mathcal{L}_k p + \mu m(p), \quad (18.34)$$

where

$$m(p) = \|c_k + A_k p\|_1,$$

and σ is a parameter to be defined below. After computing a step p_k , we choose the penalty parameter μ large enough that

$$q_\mu(0) - q_\mu(p_k) \geq \rho \mu [m(0) - m(p_k)], \quad (18.35)$$

for some parameter $\rho \in (0, 1)$. It follows from (18.34) and (18.7b) that inequality (18.35) is satisfied for

$$\mu \geq \frac{\nabla f_k^T p_k + (\sigma/2) p_k^T \nabla_{xx}^2 \mathcal{L}_k p_k}{(1 - \rho) \|c_k\|_1}. \quad (18.36)$$

If the value of μ from the previous iteration of the SQP method satisfies (18.36), it is left unchanged. Otherwise, μ is increased so that it satisfies this inequality with some margin.

The constant σ is used to handle the case in which the Hessian $\nabla_{xx}^2 \mathcal{L}_k$ is not positive definite. We define σ as

$$\sigma = \begin{cases} 1 & \text{if } p_k^T \nabla_{xx}^2 \mathcal{L}_k p_k > 0, \\ 0 & \text{otherwise.} \end{cases} \quad (18.37)$$

It is easy to verify that, if μ satisfies (18.36), this choice of σ ensures that $D(\phi_1(x_k; \mu); p_k) \leq -\rho \mu \|c_k\|_1$, so that p_k is a descent direction for the merit function ϕ_1 . This conclusion is not always valid if $\sigma = 1$ and $p_k^T \nabla_{xx}^2 \mathcal{L}_k p_k < 0$. By comparing (18.33) and (18.36) we see that, when $\sigma > 0$, the strategy based on (18.35) selects a larger penalty parameter, thus placing more weight on the reduction of the constraints. This property is advantageous if the step p_k decreases the constraints but increases the objective, for in this case the step has a better chance of being accepted by the merit function.

SECOND-ORDER CORRECTION

In Chapter 15, we showed by means of Example 15.4 that many merit functions can impede progress of an optimization algorithm, a phenomenon known as the Maratos effect. We now show that the step analyzed in that example is, in fact, produced by an SQP method.

□ **EXAMPLE 18.1** (EXAMPLE 15.4, REVISITED)

Consider problem (15.34). At the iterate $x_k = (\cos \theta, \sin \theta)^T$, let us compute a search direction p_k by solving the SQP subproblem (18.7) with $\nabla_{xx}^2 \mathcal{L}_k$ replaced by $\nabla_{xx}^2 \mathcal{L}(x^*, \lambda^*) = I$. Since

$$f_k = -\cos \theta, \quad \nabla f_k = \begin{bmatrix} 4 \cos \theta - 1 \\ 4 \sin \theta \end{bmatrix}, \quad A_k^T = \begin{bmatrix} 2 \cos \theta \\ 2 \sin \theta \end{bmatrix},$$

the quadratic subproblem (18.7) takes the form

$$\begin{aligned} \min_p \quad & (4 \cos \theta - 1)p_1 + 4 \sin \theta p_2 + \frac{1}{2}p_1^2 + \frac{1}{2}p_2^2 \\ \text{subject to} \quad & p_2 + \cot \theta p_1 = 0. \end{aligned}$$

By solving this subproblem, we obtain the direction

$$p_k = \begin{bmatrix} \sin^2 \theta \\ -\sin \theta \cos \theta \end{bmatrix}, \quad (18.38)$$

which coincides with (15.35). □

We mentioned in Section 15.4 that the difficulties associated with the Maratos effect can be overcome by means of a *second-order correction*. There are various ways of applying this technique; we describe one possible implementation next.

Suppose that the SQP method has computed a step p_k from (18.11). If this step yields an increase in the merit function ϕ_1 , a possible cause is that our linear approximations to the constraints are not sufficiently accurate. To overcome this deficiency, we could re-solve (18.11) with the linear terms $c_i(x_k) + \nabla c_i(x_k)^T p$ replaced by quadratic approximations,

$$c_i(x_k) + \nabla c_i(x_k)^T p + \frac{1}{2} p^T \nabla^2 c_i(x_k) p. \quad (18.39)$$

However, even if the Hessians of the constraints are individually available, the resulting quadratically constrained subproblem may be too difficult to solve. Instead, we evaluate the constraint values at the new point $x_k + p_k$ and make use of the following approximations. By Taylor's theorem, we have

$$c_i(x_k + p_k) \approx c_i(x_k) + \nabla c_i(x_k)^T p_k + \frac{1}{2} p_k^T \nabla^2 c_i(x_k) p_k. \quad (18.40)$$

Assuming that the (still unknown) second-order step p will not be too different from p_k , we can approximate the last term in (18.39) as follows:

$$p^T \nabla^2 c_i(x_k) p = p_k^T \nabla^2 c_i(x_k) p_k. \quad (18.41)$$

By making this substitution in (18.39) and using (18.40), we obtain the second-order correction subproblem

$$\begin{aligned} \min_p \quad & \nabla f_k^T p + \frac{1}{2} p^T \nabla_{xx}^2 \mathcal{L}_k p \\ \text{subject to} \quad & \nabla c_i(x_k)^T p + d_i = 0, \quad i \in \mathcal{E}, \\ & \nabla c_i(x_k)^T p + d_i \geq 0, \quad i \in \mathcal{I}, \end{aligned}$$

where

$$d_i = c_i(x_k + p_k) - \nabla c_i(x_k)^T p_k, \quad i \in \mathcal{E} \cup \mathcal{I}.$$

The second-order correction step requires evaluation of the constraints $c_i(x_k + p_k)$ for $i \in \mathcal{E} \cup \mathcal{I}$, and therefore it is preferable not to apply it every time the merit function increases. One strategy is to use it only if the increase in the merit function is accompanied by an increase in the constraint norm.

It can be shown that when the step p_k is generated by the SQP method (18.11) then, near a solution satisfying second-order sufficient conditions, the algorithm above takes either the full step p_k or the corrected step $p_k + \hat{p}_k$. The merit function does not interfere with the iteration, so superlinear convergence is attained, as in the local algorithm.

18.4 A PRACTICAL LINE SEARCH SQP METHOD

From the discussion in the previous section, we can see that there is a wide variety of line search SQP methods that differ in the way the Hessian approximation is computed, in the step acceptance mechanism, and in other algorithmic features. We now incorporate some of these ideas into a concrete, practical SQP algorithm for solving the nonlinear programming problem (18.10). To keep the description simple, we will not include a mechanism such as (18.12) to ensure the feasibility of the subproblem, or a second-order correction step. Rather, the search direction is obtained simply by solving the subproblem (18.11). We also assume that the quadratic program (18.11) is convex, so that we can solve it by means of the active-set method for quadratic programming (Algorithm 16.3) described in Chapter 16.

Algorithm 18.3 (Line Search SQP Algorithm).

Choose parameters $\eta \in (0, 0.5)$, $\tau \in (0, 1)$, and an initial pair (x_0, λ_0) ;

Evaluate $f_0, \nabla f_0, c_0, A_0$;

If a quasi-Newton approximation is used, choose an initial $n \times n$ symmetric positive definite Hessian approximation B_0 , otherwise compute $\nabla_{xx}^2 \mathcal{L}_0$;

repeat until a convergence test is satisfied

 Compute p_k by solving (18.11); let $\hat{\lambda}$ be the corresponding multiplier;

 Set $p_\lambda \leftarrow \hat{\lambda} - \lambda_k$;

 Choose μ_k to satisfy (18.36) with $\sigma = 1$;

 Set $\alpha_k \leftarrow 1$;

while $\phi_1(x_k + \alpha_k p_k; \mu_k) > \phi_1(x_k; \mu_k) + \eta \alpha_k D_1(\phi(x_k; \mu_k) p_k)$

 Reset $\alpha_k \leftarrow \tau_\alpha \alpha_k$ for some $\tau_\alpha \in (0, \tau]$;

end (while)

 Set $x_{k+1} \leftarrow x_k + \alpha_k p_k$ and $\lambda_{k+1} \leftarrow \lambda_k + \alpha_k p_\lambda$;

 Evaluate $f_{k+1}, \nabla f_{k+1}, c_{k+1}, A_{k+1}$, (and possibly $\nabla_{xx}^2 \mathcal{L}_{k+1}$);

 If a quasi-Newton approximation is used, set

$s_k \leftarrow \alpha_k p_k$ and $y_k \leftarrow \nabla_x \mathcal{L}(x_{k+1}, \lambda_{k+1}) - \nabla_x \mathcal{L}(x_k, \lambda_{k+1})$,

 and obtain B_{k+1} by updating B_k using a quasi-Newton formula;

end (repeat)

We can achieve significant savings in the solution of the quadratic subproblem by warm-start procedures. For example, we can initialize the working set for each QP subproblem to be the final active set from the previous SQP iteration.

We have not given particulars of the quasi-Newton approximation in Algorithm 18.3. We could use, for example, a limited-memory BFGS approach that is suitable for large-scale problems. If we use an exact Hessian $\nabla_{xx}^2 \mathcal{L}_k$, we assume that it is modified as necessary to be positive definite on the null space of the equality constraints.

Instead of a merit function, we could employ a filter (see Section 15.4) in the inner “while” loop to determine the steplength α_k . As discussed in Section 15.4, a feasibility restoration phase is invoked if a trial steplength generated by the backtracking line search is

smaller than a given threshold. Regardless of whether a merit function or filter are used, a mechanism such as second-order correction can be incorporated to overcome the Maratos effect.

18.5 TRUST-REGION SQP METHODS

Trust-region SQP methods have several attractive properties. Among them are the facts that they do not require the Hessian matrix $\nabla_{xx}^2 \mathcal{L}_k$ in (18.11) to be positive definite, they control the quality of the steps even in the presence of Hessian and Jacobian singularities, and they provide a mechanism for enforcing global convergence. Some implementations follow an IQP approach and solve an inequality-constrained subproblem, while others follow an EQP approach.

The simplest way to formulate a trust-region SQP method is to add a trust-region constraint to subproblem (18.11), as follows:

$$\min_p f_k + \nabla f_k^T p + \frac{1}{2} p^T \nabla_{xx}^2 \mathcal{L}_k p \quad (18.43a)$$

$$\text{subject to } \nabla c_i(x_k)^T p + c_i(x_k) = 0, \quad i \in \mathcal{E}, \quad (18.43b)$$

$$\nabla c_i(x_k)^T p + c_i(x_k) \geq 0, \quad i \in \mathcal{I}, \quad (18.43c)$$

$$\|p\| \leq \Delta_k. \quad (18.43d)$$

Even if the constraints (18.43b), (18.43c) are compatible, this problem may not always have a solution because of the trust-region constraint (18.43d). We illustrate this fact in Figure 18.1 for a problem that contains only one equality constraint whose linearization is represented by the solid line. In this example, any step p that satisfies the linearized constraint must lie outside the trust region, which is indicated by the circle of radius Δ_k . As we see from this example, a consistent system of equalities and inequalities may not have a solution if we restrict the norm of the solution.

To resolve the possible conflict between the linear constraints (18.43b), (18.43c) and the trust-region constraint (18.43d), it is not appropriate simply to increase Δ_k until the set of steps p satisfying the linear constraints intersects the trust region. This approach would defeat the purpose of using the trust region in the first place as a way to define a region within which we trust the model (18.43a)–(18.43c) to accurately reflect the behavior of the objective and constraint functions. Analytically, it would harm the convergence properties of the algorithm.

A more appropriate viewpoint is that there is no reason to satisfy the linearized constraints exactly at every step; rather, we should aim to improve the feasibility of these constraints at each step and to satisfy them exactly only if the trust-region constraint permits it. This point of view is the basis of the three classes of methods discussed in this section: relaxation methods, penalty methods, and filter methods.

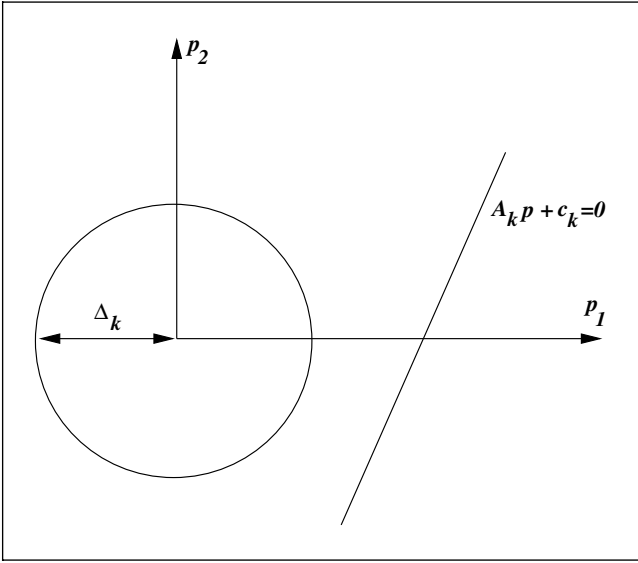


Figure 18.1 Inconsistent constraints in trust-region model.

A RELAXATION METHOD FOR EQUALITY-CONSTRAINED OPTIMIZATION

We describe this method in the context of the equality-constrained optimization problem (18.1); its extension to general nonlinear programs is deferred to Chapter 19 because it makes use of interior-point techniques. (Active-set extensions of the relaxation approach have been proposed, but have not been fully explored.)

At the iterate x_k , we compute the SQP step by solving the subproblem

$$\min_p \quad f_k + \nabla f_k^T p + \frac{1}{2} p^T \nabla_{xx}^2 \mathcal{L}_k p \quad (18.44a)$$

$$\text{subject to} \quad A_k p + c_k = r_k, \quad (18.44b)$$

$$\|p\|_2 \leq \Delta_k. \quad (18.44c)$$

The choice of the relaxation vector r_k requires careful consideration, as it impacts the efficiency of the method. Our goal is to choose r_k as the smallest vector such that (18.44b), (18.44c) are consistent for some reduced value of trust-region radius Δ_k . To do so, we first solve the subproblem

$$\min_v \quad \|A_k v + c_k\|_2^2 \quad (18.45a)$$

$$\text{subject to} \quad \|v\|_2 \leq 0.8\Delta_k. \quad (18.45b)$$

Denoting the solution of this subproblem by v_k , we define

$$r_k = A_k v_k + c_k. \quad (18.46)$$

We now compute the step p_k by solving (18.44), define the new iterate $x_{k+1} = x_k + p_k$, and obtain new multiplier estimates λ_{k+1} using the least squares formula (18.21). Note that the constraints (18.44b), (18.44c) are consistent because they are satisfied by the vector $p = v_k$.

At first glance, this approach appears to be impractical because problems (18.44) and (18.45) are not particularly easy to solve, especially when $\nabla_{xx}^2 \mathcal{L}_k$ is indefinite. Fortunately, we can design efficient procedures for computing useful *inexact* solutions of these problems.

We solve the auxiliary subproblem (18.45) by the dogleg method described in Chapter 4. This method requires a Cauchy step p^u , which is the minimizer of the objective (18.45a) along the direction $-A_k^T c_k$, and a “Newton step” p^b , which is the unconstrained minimizer of (18.45a). Since the Hessian in (18.45a) is singular, there are infinitely many possible choices of p^b , all of which satisfy $A_k p^b + c_k = 0$. We choose the one with smallest Euclidean norm by setting

$$p^b = -A_k^T [A_k A_k^T]^{-1} c_k.$$

We now take v_k to be the minimizer of (18.45a) along the path defined by p^u , p^b , and the formula (4.16).

The preferred technique for computing an approximate solution p_k of (18.44) is the projected conjugate gradient method of Algorithm 16.2. We apply this algorithm to the equality-constrained quadratic program (18.44a)–(18.44b), monitoring satisfaction of the trust-region constraint (18.44c) and stopping if the boundary of this region is reached or if negative curvature is detected; see Section 7.1. Algorithm 16.2 requires a feasible starting point, which may be chosen as v_k .

A merit function that fits well with this approach is the nonsmooth ℓ_2 function $\phi_2(x; \mu) = f(x) + \mu \|c(x)\|_2$. We model it by means of the function

$$q_\mu(p) = f_k + \nabla f_k^T p + \frac{1}{2} p^T \nabla_{xx}^2 \mathcal{L}_k p + \mu m(p), \quad (18.47)$$

where

$$m(p) = \|c_k + A_k p\|_2,$$

(see (18.34)). We choose the penalty parameter large enough that inequality (18.35) is satisfied. To judge the acceptability of a step p_k , we monitor the ratio

$$\rho_k = \frac{\text{ared}_k}{\text{pred}_k} = \frac{\phi_2(x_k, \mu) - \phi_2(x_k + p_k, \mu)}{q_\mu(0) - q_\mu(p_k)}. \quad (18.48)$$

We can now give a description of this trust-region SQP method for the equality-constrained optimization problem (18.1).

Algorithm 18.4 (Byrd–Omojokun Trust-Region SQP Method).

Choose constants $\epsilon > 0$ and $\eta, \gamma \in (0, 1)$;
 Choose starting point x_0 , initial trust region $\Delta_0 > 0$;
for $k = 0, 1, 2, \dots$
 Compute $f_k, c_k, \nabla f_k, A_k$;
 Compute multiplier estimates $\hat{\lambda}_k$ by (18.21);
 if $\|\nabla f_k - A_k^T \hat{\lambda}_k\|_\infty < \epsilon$ **and** $\|c_k\|_\infty < \epsilon$
 stop with approximate solution x_k ;
 Solve normal subproblem (18.45) for v_k and compute r_k from (18.46);
 Compute $\nabla_{xx}^2 \mathcal{L}_k$ or a quasi-Newton approximation;
 Compute p_k by applying the projected CG method to (18.44);
 Choose μ_k to satisfy (18.35);
 Compute $\rho_k = \text{ared}_k / \text{pred}_k$;
 if $\rho_k > \eta$
 Set $x_{k+1} = x_k + p_k$;
 Choose Δ_{k+1} to satisfy $\Delta_{k+1} \geq \Delta_k$;
 else
 Set $x_{k+1} = x_k$;
 Choose Δ_{k+1} to satisfy $\Delta_{k+1} \leq \gamma \|p_k\|$;
end (for).

A second-order correction can be added to avoid the Maratos effect. Beyond the cost of evaluating the objective function f and constraints c , the main costs of this algorithm lie in the projected CG iteration, which requires products of the Hessian $\nabla_{xx}^2 \mathcal{L}_k$ with vectors, and in the factorization and backsolves with the projection matrix (16.32); see Section 16.3.

 ℓ_1 QP (SEQUENTIAL ℓ_1 QUADRATIC PROGRAMMING)

In this approach we move the linearized constraints (18.43b), (18.43c) into the objective of the quadratic program, in the form of an ℓ_1 penalty term, to obtain the following subproblem:

$$\begin{aligned}
 \min_p \quad q_\mu(p) \stackrel{\text{def}}{=} & f_k + \nabla f_k^T p + \frac{1}{2} p^T \nabla_{xx}^2 \mathcal{L}_k p + \mu \sum_{i \in \mathcal{E}} |c_i(x_k) + \nabla c_i(x_k)^T p| \\
 & + \mu \sum_{i \in \mathcal{I}} [c_i(x_k) + \nabla c_i(x_k)^T p]^-
 \end{aligned} \tag{18.49}$$

subject to $\|p\|_\infty \leq \Delta_k$,

for some penalty parameter μ , where we use the notation $[y]^- = \max\{0, -y\}$. Introducing slack variables v, w, t , we can reformulate this problem as follows:

$$\min_{p, v, w, t} \quad f_k + \nabla f_k^T p + \frac{1}{2} p^T \nabla_{xx}^2 \mathcal{L}_k p + \mu \sum_{i \in \mathcal{E}} (v_i + w_i) + \mu \sum_{i \in \mathcal{I}} t_i \quad (18.50a)$$

$$\text{s.t.} \quad \nabla c_i(x_k)^T p + c_i(x_k) = v_i - w_i, \quad i \in \mathcal{E}, \quad (18.50b)$$

$$\nabla c_i(x_k)^T p + c_i(x_k) \geq -t_i, \quad i \in \mathcal{I}, \quad (18.50c)$$

$$v, w, t \geq 0, \quad (18.50d)$$

$$\|p\|_\infty \leq \Delta_k. \quad (18.50e)$$

This formulation is simply a linearization of the elastic-mode formulation (18.12) with the addition of a trust-region constraint.

The constraints of this problem are always consistent. Since the trust region has been defined using the ℓ_∞ norm, (18.50) is a smooth quadratic program that can be solved by means of a quadratic programming algorithm. Warm-start strategies can significantly reduce the solution time of (18.50) and are invariably used in practical implementations.

It is natural to use the ℓ_1 merit function

$$\phi_1(x; \mu) = f(x) + \mu \sum_{i \in \mathcal{E}} |c_i(x)| + \mu \sum_{i \in \mathcal{I}} [c_i(x)]^- \quad (18.51)$$

to determine step acceptance. In fact, the function q_μ defined in (18.49) can be viewed as a model of $\phi_1(x, \mu)$ at x_k in which we approximate each constraint function c_i by its linearization, and replace f by a quadratic function whose curvature term includes information from both objective and constraints.

After computing the step p_k from (18.50), we determine the ratio ρ_k via (18.48), using the merit function ϕ_1 and defining q_μ by (18.49). The step is accepted or rejected according to standard trust-region rules, as implemented in Algorithm 18.4. A second-order correction step can be added to prevent the occurrence of the Maratos effect.

The $S\ell_1$ QP approach has several attractive properties. Not only does the formulation (18.49) overcome the possible inconsistency among the linearized constraints, but it also ensures that the trust-region constraint can always be satisfied. Further, the matrix $\nabla_{xx}^2 \mathcal{L}_k$ can be used without modification in subproblem (18.50) or else can be replaced by a quasi-Newton approximation. There is no requirement for it to be positive definite.

This choice of the penalty parameter μ plays an important role in the efficiency of this method. Unlike the SQP methods described above, which use a penalty function only to determine the acceptability of a trial point, the step p_k of the $S\ell_1$ QP algorithm depends on μ . Values of μ that are too small can lead the algorithm away from the solution (Section 17.2), while excessively large values can result in slow progress. To obtain good

practical performance over a range of applications, the value of μ must be chosen carefully at each iteration; see Algorithm 18.5 below.

SEQUENTIAL LINEAR-QUADRATIC PROGRAMMING (SLQP)

The SQP methods discussed above require the solution of a general (inequality-constrained) quadratic problem at each iteration. The cost of solving this subproblem imposes a limit on the size of problems that can be solved in practice. In addition, the incorporation of (indefinite) second derivative information in SQP methods has proved to be difficult [147].

The sequential linear-quadratic programming (SLQP) method attempts to overcome these concerns by computing the step in two stages, each of which scales well with the number of variables. First, a linear program (LP) is solved to identify a working set \mathcal{W} . Second, there is an equality-constrained quadratic programming (EQP) phase in which the constraints in the working set \mathcal{W} are imposed as equalities. The total step of the algorithm is a combination of the steps obtained in the linear programming and equality-constrained phases, as we now discuss.

In the LP phase, we would like to solve the problem

$$\min_p f_k + \nabla f_k^T p \quad (18.52a)$$

$$\text{subject to } c_i(x_k) + \nabla c_i(x_k)^T p = 0, \quad i \in \mathcal{E}, \quad (18.52b)$$

$$c_i(x_k) + \nabla c_i(x_k)^T p \geq 0, \quad i \in \mathcal{I}, \quad (18.52c)$$

$$\|p\|_\infty \leq \Delta_k^{\text{LP}}, \quad (18.52d)$$

which differs from the standard SQP subproblem (18.43) only in that the second-order term in the objective has been omitted and that an ℓ_∞ norm is used to define the trust region. Since the constraints of (18.52) may be inconsistent, we solve instead the ℓ_1 penalty reformulation of (18.52) defined by

$$\begin{aligned} \min_p \quad l_\mu(p) &\stackrel{\text{def}}{=} f_k + \nabla f_k^T p + \mu \sum_{i \in \mathcal{E}} |c_i(x_k) + \nabla c_i(x_k)^T p| \\ &\quad + \mu \sum_{i \in \mathcal{I}} [c_i(x_k) + \nabla c_i(x_k)^T p]^- \end{aligned} \quad (18.53a)$$

$$\text{subject to } \|p\|_\infty \leq \Delta_k^{\text{LP}}. \quad (18.53b)$$

By introducing slack variables as in (18.50), we can reformulate (18.53) as an LP. The solution of (18.53), which we denote by p^{LP} , is computed by the simplex method (Chapter 13). From this solution we obtain the following explicit estimate of the optimal active set:

$$\mathcal{A}_k(p^{\text{LP}}) = \{i \in \mathcal{E} \mid c_i(x_k) + \nabla c_i(x_k)^T p^{\text{LP}} = 0\} \cup \{i \in \mathcal{I} \mid c_i(x_k) + \nabla c_i(x_k)^T p^{\text{LP}} = 0\}.$$

Likewise, we define the set \mathcal{V}_k of violated constraints as

$$\mathcal{V}_k(p^{\text{LP}}) = \{i \in \mathcal{E} \mid c_i(x_k) + \nabla c_i(x_k)^T p^{\text{LP}} \neq 0\} \cup \{i \in \mathcal{I} \mid c_i(x_k) + \nabla c_i(x_k)^T p^{\text{LP}} < 0\}.$$

We define the working set \mathcal{W}_k as some linearly independent subset of the active set $\mathcal{A}_k(p^{\text{LP}})$. To ensure that the algorithm makes progress on the penalty function ϕ_1 , we define the *Cauchy step*,

$$p^{\text{C}} = \alpha^{\text{LP}} p^{\text{LP}}, \quad (18.54)$$

where $\alpha^{\text{LP}} \in (0, 1]$ is a steplength that provides sufficient decrease in the model q_μ defined in (18.49).

Given the working set \mathcal{W}_k , we now solve an equality-constrained quadratic program (EQP) treating the constraints in \mathcal{W}_k as equalities and ignoring all others. We thus obtain the subproblem

$$\min_p \quad f_k + \frac{1}{2} p^T \nabla_{xx}^2 \mathcal{L}_k p + \left(\nabla f_k + \mu_k \sum_{i \in \mathcal{V}_k} \gamma_i \nabla c_i(x_k) \right)^T p \quad (18.55a)$$

$$\text{subject to} \quad c_i(x_k) + \nabla c_i(x_k)^T p = 0, \quad i \in \mathcal{E} \cap \mathcal{W}_k, \quad (18.55b)$$

$$c_i(x_k) + \nabla c_i(x_k)^T p = 0, \quad i \in \mathcal{I} \cap \mathcal{W}_k, \quad (18.55c)$$

$$\|p\|_2 \leq \Delta_k, \quad (18.55d)$$

where γ_i is the algebraic sign of the i -th violated constraint. Note that the trust region (18.55d) is spherical, and that Δ_k is distinct from the trust-region radius Δ_k^{LP} used in (18.53b). Problem (18.55) is solved for the vector p^{Q} by applying the projected conjugated gradient procedure of Algorithm 16.2, handling the trust-region constraint by Steihaug's strategy (Algorithm 7.2). The total step p_k of the SLQP method is given by

$$p_k = p^{\text{C}} + \alpha^{\text{Q}}(p^{\text{Q}} - p^{\text{C}}),$$

where $\alpha^{\text{Q}} \in [0, 1]$ is a steplength that approximately minimizes the model q_μ defined in (18.49).

The trust-region radius Δ_k for the EQP phase is updated using standard trust-region update strategies. The choice of radius Δ_{k+1}^{LP} for the LP phase is more delicate, since it influences our guess of the optimal active set. The value of Δ_{k+1}^{LP} should be set to be a little larger than the total step p_k , subject to some other restrictions [49]. The multiplier estimates λ_k used in the Hessian $\nabla_{xx}^2 \mathcal{L}_k$ are least squares estimates (18.21) using the working set \mathcal{W}_k , and modified so that $\lambda_i \geq 0$ for $i \in \mathcal{I}$.

An appealing feature of the SLQP algorithm is that established techniques for solving large-scale versions of the LP and EQP subproblems are readily available. High quality LP

software is capable of solving problems with very large numbers of variables and constraints, while the solution of the EQP subproblem can be performed efficiently using the projected conjugate gradient method.

A TECHNIQUE FOR UPDATING THE PENALTY PARAMETER

We have mentioned that penalty methods such as Sl_1 QP and SLQP can be sensitive to the choice of the penalty parameter μ . We now discuss a procedure for choosing μ that has proved to be effective in practice and is supported by global convergence guarantees. The goal is to choose μ small enough to avoid an unnecessary imbalance in the merit function, but large enough to cause the step to make sufficient progress in linearized feasibility at each iteration. We present this procedure in the context of the Sl_1 QP method and then describe its extension to the SLQP approach.

We define a piecewise linear model of constraint violation at a point x_k by

$$m_k(p) = \sum_{i \in \mathcal{E}} |c_i(x_k) + \nabla c_i(x_k)^T p| + \sum_{i \in \mathcal{I}} [c_i(x_k) + \nabla c_i(x_k)^T p]^{-}, \quad (18.56)$$

so that the objective of the SQP subproblem (18.49) can be written as

$$q_\mu(p) = f_k + \nabla f_k^T p + \frac{1}{2} p^T \nabla_{xx}^2 \mathcal{L}_k p + \mu m_k(p). \quad (18.57)$$

We begin by solving the QP subproblem (18.49) (or equivalently, (18.50)) using the previous value μ_{k-1} of the penalty parameter. If the constraints (18.50b), (18.50c) are satisfied with the slack variables v_i , w_i , t_i all equal to zero (that is, $m_k(p_k) = 0$), then the current value of μ is adequate, and we set $\mu_k = \mu_{k-1}$. This is the felicitous case in which we can achieve linearized feasibility with a step p_k that is no longer in norm than the trust-region radius.

If $m_k(p) > 0$, on the other hand, it may be appropriate to increase the penalty parameter. The question is: by how much? To obtain a reference value, we re-solve the QP (18.49) using an “infinite” value of μ , by which we mean that the objective function in (18.49) is replaced by $m_k(p)$. After computing the new step, which we denote by p_∞ , two outcomes are possible. If $m_k(p_\infty) = 0$, meaning that the linearized constraints are feasible within the trust region, we choose $\mu_k > \mu_{k-1}$ such that $m_k(p_k) = 0$. Otherwise, if $m_k(p_\infty) > 0$, we choose $\mu_k \geq \mu_{k-1}$ such that the reduction in m_k caused by the step p_k is at least a fraction of the (optimal) reduction given by p_∞ .

The selection of $\mu_k > \mu_{k-1}$ is achieved in all cases by successively increasing the current trial value of μ (by a factor of 10, say) and re-solving the quadratic program (18.49). To describe this strategy more precisely, we write the solution of the QP problem (18.49) as $p(\mu)$ to stress its dependence on the penalty parameter. Likewise, p_∞ denotes the minimizer of $m_k(p)$ subject to the trust-region constraint (18.50e). The following algorithm describes the selection of the penalty parameter μ_k and the computation of the Sl_1 QP step p_k .

Algorithm 18.5 (Penalty Update and Step Computation).

Initial data: $x_k, \mu_{k-1} > 0, \Delta_k > 0$, and parameters $\epsilon_1, \epsilon_2 \in (0, 1)$.

Solve the subproblem (18.50) with $\mu = \mu_{k-1}$ to obtain $p(\mu_{k-1})$;

if $m_k(p(\mu_{k-1})) = 0$

Set $\mu^+ \leftarrow \mu_{k-1}$;

else

Compute p_∞ ;

if $m_k(p_\infty) = 0$

Find $\mu^+ > \mu_{k-1}$ such that $m_k(p(\mu^+)) = 0$;

else

Find $\mu^+ \geq \mu_{k-1}$ such that

$$m_k(0) - m_k(p(\mu^+)) \geq \epsilon_1[m_k(0) - m_k(p_\infty)];$$

end(if)

end(if)

Increase μ^+ if necessary to satisfy

$$q_{\mu^+}(0) - q_{\mu^+}(p(\mu^+)) \geq \epsilon_2 \mu^+[m_k(0) - m_k(p(\mu^+))];$$

Set $\mu_k \leftarrow \mu^+$ and $p_k \leftarrow p(\mu^+)$.

(Note that the inequality in the penultimate line is the same as condition (18.35).) Although Algorithm 18.5 requires the solution of some additional quadratic programs, we hope to reduce the total number of iterations (and the total number of QP solves) by identifying an appropriate penalty parameter value more quickly than rules based on feasibility monitoring (see Framework 17.2).

Numerical experience indicates that these savings occur when an adaptation of Algorithm 18.5 is used in the SLQP method. This adaptation is obtained simply by setting $\nabla_{xx}^2 \mathcal{L}_k = 0$ in the definition (18.49) of q_μ and applying Algorithm 18.5 to determine μ and to compute the LP step p^{lp} . The extra LP solves required by Algorithm 18.5 in this case are typically inexpensive, requiring relatively few simplex iterations, because we can use warm-start information from LPs solved earlier, with different values of the penalty parameter.

18.6 NONLINEAR GRADIENT PROJECTION

In Section 16.7, we discussed the gradient projection method for bound constrained quadratic programming. It is not difficult to extend this method to the problem

$$\min f(x) \quad \text{subject to} \quad l \leq x \leq u, \quad (18.58)$$

where f is a nonlinear function and l and u are vectors of lower and upper bounds, respectively.

We begin by describing a line search approach. At the current iterate x_k , we form the quadratic model

$$q_k(x) = f_k + \nabla f_k^T (x - x_k) + \frac{1}{2} (x - x_k)^T B_k (x - x_k), \quad (18.59)$$

where B_k is a positive definite approximation to $\nabla^2 f(x_k)$. We then use the gradient projection method for quadratic programming (Algorithm 16.5) to find an approximate solution \hat{x} of the subproblem

$$\min q_k(x) \quad \text{subject to} \quad l \leq x \leq u. \quad (18.60)$$

The search direction is defined as $p_k = \hat{x} - x_k$ and the new iterate is given by $x_{k+1} = x_k + \alpha_k p_k$, where the steplength α_k is chosen to satisfy

$$f(x_k + \alpha_k p_k) \leq f(x_k) + \eta \alpha_k \nabla f_k^T p_k$$

for some parameter $\eta \in (0, 1)$.

To see that the search direction p_k is indeed a descent direction for the objective function, we use the properties of Algorithm 16.5, as discussed in Section 16.7. Recall that this method searches along a piecewise linear path—the projected steepest descent path—for the Cauchy point x^c , which minimizes q_k along this path. It then identifies the components of x that are at their bounds and holds these components constant while performing an unconstrained minimization of q_k over the remaining components to obtain the approximate solution \hat{x} of the subproblem (18.60).

The Cauchy point x^c satisfies $q_k(x^c) < q_k(x_k)$ if the projected gradient is nonzero. Since Algorithm 16.5 produces a subproblem solution \hat{x} with $q_k(\hat{x}) \leq q_k(x^c)$, we have

$$f_k = q_k(x_k) > q_k(x^c) \geq q_k(\hat{x}) = f_k + \nabla f_k^T p_k + \frac{1}{2} p_k^T B_k p_k.$$

This inequality implies that $\nabla f_k^T p_k < 0$, since B_k is assumed to be positive definite.

We now consider a trust-region gradient projection method for solving (18.58). We begin by forming the quadratic model (18.59), but since there is no requirement for q_k to be convex, we can define B_k to be the Hessian $\nabla^2 f(x_k)$ or a quasi-Newton approximation obtained from the BFGS or SR1 formulas. The step p_k is obtained by solving the subproblem

$$\min q_k(x) \quad \text{subject to} \quad \{l \leq x \leq u, \quad \|x - x_k\|_\infty \leq \Delta_k\}, \quad (18.61)$$

for some $\Delta_k > 0$. This problem can be posed as a bound-constrained quadratic program as follows:

$$\min q_k(x) \quad \text{subject to} \quad \max(l, x_k - \Delta_k e) \leq x \leq \min(u, x_k + \Delta_k e),$$

where $e = (1, 1, \dots, 1)^T$. Algorithm 16.5 can be used to solve this subproblem. The step p_k is accepted or rejected following standard trust-region strategies, and the radius Δ_k is updated according to the agreement between the change in f and the change in q_k produced by the step p_k ; see Chapter 4.

The two gradient projection methods just outlined require solution of an inequality-constrained quadratic subproblem at each iteration, and so are formally IQP methods. They can, however, be viewed also as EQP methods because of their use of Algorithm 16.5 in solving the subproblem. This algorithm first identifies a working set (by finding the Cauchy point) and then solves an equality-constrained subproblem (by fixing the working-set constraints at their bounds). For large problems, it is efficient to perform the subspace minimization (16.74) by using the conjugate gradient method. A preconditioner is sometimes needed to make this approach practical; the most popular choice is the incomplete (and modified) Cholesky factorization outlined in Algorithm 7.3.

The gradient projection approach can be extended in principle to more general (linear or convex) constraints. Practical implementations are however limited to the bound constrained problem (18.58) because of the high cost of computing projections onto general constraint sets.

18.7 CONVERGENCE ANALYSIS

Numerical experience has shown that the SQP and SLQP methods discussed in this chapter often converge to a solution from remote starting points. Hence, there has been considerable interest in understanding what drives the iterates toward a solution and what can cause the algorithms to fail. These global convergence studies have been valuable in improving the design and implementation of algorithms.

Some early results make strong assumptions, such as boundedness of multipliers, well posedness of the subproblem (18.11), and regularity of constraint Jacobians. More recent studies relax many of these assumptions with the goal of understanding both the successful and unsuccessful outcomes of the iteration. We now state a classical global convergence result that gives conditions under which a standard SQP algorithm always identifies a KKT point of the nonlinear program.

Consider an SQP method that computes a search direction p_k by solving the quadratic program (18.11). We assume that the Hessian $\nabla_{xx}^2 \mathcal{L}_k$ is replaced in (18.11a) by some symmetric and positive definite approximation B_k . The new iterate is defined as $x_{k+1} + \alpha_k p_k$, where α_k is computed by a backtracking line search, starting from the unit steplength, and terminating when

$$\phi_1(x_k + \alpha_k p_k; \mu) \leq \phi_1(x_k; \mu) - \eta \alpha_k (q_\mu(0) - q_\mu(p_k)),$$

where $\eta \in (0, 1)$, with ϕ_1 defined as in (18.51) and q_μ defined as in (18.49). To establish the convergence result, we assume that each quadratic program (18.11) is feasible and

determines a bounded solution p_k . We also assume that the penalty parameter μ is fixed for all k and sufficiently large.

Theorem 18.3.

Suppose that the SQP algorithm just described is applied to the nonlinear program (18.10). Suppose that the sequences $\{x_k\}$ and $\{x_k + p_k\}$ are contained in a closed, bounded, convex region of \mathbb{R}^n in which f and c_i have continuous first derivatives. Suppose that the matrices B_k and multipliers are bounded and that μ satisfies $\mu \geq \|\lambda_k\|_\infty + \rho$ for all k , where ρ is a positive constant. Then all limit points of the sequence $\{x_k\}$ are KKT points of the nonlinear program (18.10).

The conclusions of the theorem are quite satisfactory, but the assumptions are somewhat restrictive. For example, the condition that the sequence $\{x_k + p_k\}$ stays within a bounded set rules out the case in which the Hessians B_k or constraint Jacobians become ill conditioned. Global convergence results that are established under more realistic conditions are surveyed by Conn, Gould, and Toint [74]. An example of a result of this type is Theorem 19.2. Although this theorem is established for a nonlinear interior-point method, similar results can be established for trust-region SQP methods.

RATE OF CONVERGENCE

We now derive conditions that guarantee the local convergence of SQP methods, as well as conditions that ensure a superlinear rate of convergence. For simplicity, we limit our discussion to Algorithm 18.1 for equality-constrained optimization, and consider both exact Hessian and quasi-Newton versions. The results presented here can be applied to algorithms for inequality-constrained problems once the active set has settled at its final optimal value (see Theorem 18.1).

We begin by listing a set of assumptions on the problem that will be useful in this section.

Assumptions 18.2.

The point x^ is a local solution of problem (18.1) at which the following conditions hold.*

- (a) *The functions f and c are twice differentiable in a neighborhood of x^* with Lipschitz continuous second derivatives.*
- (b) *The linear independence constraint qualification (Definition 12.4) holds at x^* . This condition implies that the KKT conditions (12.34) are satisfied for some vector of multipliers λ^* .*
- (c) *The second-order sufficient conditions (Theorem 12.6) hold at (x^*, λ^*) .*

We consider first an SQP method that uses exact second derivatives.

Theorem 18.4.

Suppose that Assumptions 18.2 hold. Then, if (x_0, λ_0) is sufficiently close to (x^, λ^*) , the pairs (x_k, λ_k) generated by Algorithm 18.1 converge quadratically to (x^*, λ^*) .*

The proof follows directly from Theorem 11.2, since we know that Algorithm 18.1 is equivalent to Newton's method applied to the nonlinear system $F(x, \lambda) = 0$, where F is defined by (18.3).

We turn now to quasi-Newton variants of Algorithm 18.1, in which the Lagrangian Hessian $\nabla_{xx}^2 \mathcal{L}(x_k, \lambda_k)$ is replaced by a quasi-Newton approximation B_k . We discussed in Section 18.3 algorithms that used approximations to the full Hessian, and also reduced-Hessian methods that maintained approximations to the projected Hessian $Z_k^T \nabla_{xx}^2 \mathcal{L}(x_k, \lambda_k) Z_k$. As in the earlier discussion, we take Z_k to be the $n \times (n - m)$ matrix whose columns span the null space of A_k , assuming in addition that the columns of Z_k are orthonormal; see (15.22).

If we multiply the first block row of the KKT system (18.9) by Z_k , we obtain

$$Z_k^T \nabla_{xx}^2 \mathcal{L}_k p_k = -Z_k^T \nabla f_k. \quad (18.62)$$

This equation, together with the second block row $A_k p_k = -c_k$ of (18.9), is sufficient to determine fully the value of p_k when x_k and λ_k are not too far from their optimal values. In other words, only the projection of the Hessian $Z_k^T \nabla_{xx}^2 \mathcal{L}_k$ is significant; the remainder of $\nabla_{xx}^2 \mathcal{L}_k$ (its projection onto the range space of A_k^T) does not play a role in determining p_k .

By multiplying (18.62) by Z_k , and defining the following matrix P_k , which projects onto the null space of A_k :

$$P_k = I - A_k^T [A_k A_k^T]^{-1} A_k = Z_k Z_k^T,$$

we can rewrite (18.62) equivalently as follows:

$$P_k \nabla_{xx}^2 \mathcal{L}_k p_k = -P_k \nabla f_k.$$

The discussion above, together with Theorem 18.4, suggests that a quasi-Newton method will be locally convergent if the quasi-Newton matrix B_k is chosen so that $P_k B_k$ is a reasonable approximation of $P_k \nabla_{xx}^2 \mathcal{L}_k$, and that it will be superlinearly convergent if $P_k B_k$ approximates $P_k \nabla_{xx}^2 \mathcal{L}_k$ well. To make the second statement more precise, we present a result that can be viewed as an extension of characterization of superlinear convergence (Theorem 3.6) to the equality-constrained case. In the following discussion, $\nabla_{xx}^2 \mathcal{L}_*$ denotes $\nabla_{xx}^2 \mathcal{L}(x^*, \lambda^*)$.

Theorem 18.5.

Suppose that Assumptions 18.2 hold and that the iterates x_k generated by Algorithm 18.1 with quasi-Newton approximate Hessians B_k converge to x^ . Then x_k converges superlinearly if and only if the Hessian approximation B_k satisfies*

$$\lim_{k \rightarrow \infty} \frac{\|P_k(B_k - \nabla_{xx}^2 \mathcal{L}_*)(x_{k+1} - x_k)\|}{\|x_{k+1} - x_k\|} = 0. \quad (18.63)$$

We can apply this result to the quasi-Newton updating schemes discussed earlier in this chapter, beginning with the full BFGS approximation based on (18.13). To guarantee that the BFGS approximation is always well defined, we make the (strong) assumption that the Hessian of the Lagrangian is positive definite at the solution.

Theorem 18.6.

Suppose that Assumptions 18.2 hold. Assume also that $\nabla_{xx}^2 \mathcal{L}_$ and B_0 are symmetric and positive definite. If $\|x_0 - x^*\|$ and $\|B_0 - \nabla_{xx}^2 \mathcal{L}_*\|$ are sufficiently small, the iterates x_k generated by Algorithm 18.1 with BFGS Hessian approximations B_k defined by (18.13) and (18.16) (with $r_k = s_k$) satisfy the limit (18.63). Therefore, the iterates x_k converge superlinearly to x^* .*

For the damped BFGS updating strategy given in Procedure 18.2, we can show that the rate of convergence is R-superlinear (not the usual Q-superlinear rate; see the Appendix).

We now consider reduced-Hessian SQP methods that update an approximation M_k to $Z_k^T \nabla_{xx}^2 \mathcal{L}_k Z_k$. From the definition of P_k , we see that $Z_k M_k Z_k^T$ can be considered as an approximation to the *two-sided* projection $P_k \nabla_{xx}^2 \mathcal{L}_k P_k$. Since reduced-Hessian methods do not approximate the one-sided projection $P_k \nabla_{xx}^2 \mathcal{L}_k$, we cannot expect (18.63) to hold. For these methods, we can state a condition for superlinear convergence by writing (18.63) as

$$\lim_{k \rightarrow \infty} \left[\frac{P_k(B_k - \nabla_{xx}^2 \mathcal{L}_*) P_k(x_{k+1} - x_k)}{\|x_{k+1} - x_k\|} + \frac{P_k(B_k - \nabla_{xx}^2 \mathcal{L}_*)(I - P_k)(x_{k+1} - x_k)}{\|x_{k+1} - x_k\|} \right] = 0, \quad (18.64)$$

and defining $B_k = Z_k M_k Z_k^T$. The following result shows that it is necessary only for the first term in (18.64) to go to zero to obtain a weaker form of superlinear convergence, namely, two-step superlinear convergence.

Theorem 18.7.

Suppose that Assumption 18.2(a) holds and that the matrices B_k are bounded. Assume also that the iterates x_k generated by Algorithm 18.1 with approximate Hessians B_k converge to x^ , and that*

$$\lim_{k \rightarrow \infty} \frac{\|P_k(B_k - \nabla_{xx}^2 \mathcal{L}_*) P_k(x_{k+1} - x_k)\|}{\|x_{k+1} - x_k\|} = 0. \quad (18.65)$$

Then the sequence $\{x_k\}$ converges to x^ two-step superlinearly, that is,*

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

In a reduced-Hessian method that uses BFGS updating, the iteration is $x_{k+1} = x_k + Y_k p_Y + Z_k p_Z$, where p_Y and p_Z are given by (18.19a), (18.23) (with $(Z_k^T \nabla_{xx}^2 \mathcal{L}_k Z_k)$ replaced by M_k). The reduced-Hessian approximation M_k is updated by the BFGS formula using

the correction vectors (18.26), and the initial approximation M_0 is symmetric and positive definite. If we make the assumption that the null space bases Z_k used to define the correction vectors (18.26) vary smoothly, then we can apply Theorem 18.7 to show that x_k converges two-step superlinearly.

18.8 PERSPECTIVES AND SOFTWARE

SQP methods are most efficient if the number of active constraints is nearly as large as the number of variables, that is, if the number of free variables is relatively small. They require few evaluations of the functions, in comparison with augmented Lagrangian methods, and can be more robust on badly scaled problems than the nonlinear interior-point methods described in the next chapter. It is not known at present whether the IQP or EQP approach will prove to be more effective for large problems. Current research focuses on widening the class of problems that can be solved with SQP and SLQP approaches.

Two established SQP software packages are SNOPT [128] and FILTERSQP [105]. The former code follows a line search approach, while the latter implements a trust-region strategy using a filter for step acceptance. The SLQP approach of Section 18.5 is implemented in KNITRO/ACTIVE [49]. All three packages include mechanisms to ensure that the subproblems are always feasible and to guard against rank-deficient constraint Jacobians. SNOPT uses the penalty (or elastic) mode (18.12), which is invoked if the SQP subproblem is infeasible or if the Lagrange multiplier estimates become very large in norm. FILTERSQP includes a feasibility restoration phase that, in addition to promoting convergence, provides rapid identification of convergence to infeasible points. KNITRO/ACTIVE implements a penalty method using the update strategy of Algorithm 18.5.

There is no established implementation of the $S\ell_1$ QP approach, but prototype implementations have shown promise. The CONOPT [9] package implements a generalized reduced gradient method as well as an SQP method.

Quasi-Newton approximations to the Hessian of the Lagrangian $\nabla_{xx}^2 \mathcal{L}_k$ are often used in practice. BFGS updating is generally less effective for constrained problems than in the unconstrained case because of the requirement of maintaining a positive definite approximation to an underlying matrix that often does not have this property. Nevertheless, the BFGS and limited-memory BFGS approximations implemented in SNOPT and KNITRO perform adequately in practice. KNITRO also offers an SR1 option that may be more effective than the BFGS option, but the question of how best to implement full quasi-Newton approximations for constrained optimization requires further investigation. The RSQP package [13] implements an SQP method that maintains a quasi-Newton approximation to the reduced Hessian.

The Maratos effect, if left unattended, can significantly slow optimization algorithms that use nonsmooth merit functions or filters. However, selective application of second-order correction steps adequately resolves the difficulties in practice.

Trust-region implementations of the gradient projection method include TRON [192] and LANCELOT [72]. Both codes use a conjugate gradient iteration to perform the subspace minimization and apply an incomplete Cholesky preconditioner. Gradient projection methods in which the Hessian approximation is defined by limited-memory BFGS updating are implemented in LBFGS-B [322] and BLMVM [17]. The properties of limited-memory BFGS matrices can be exploited to perform the projected gradient search and subspace minimization efficiently. SPG [23] implements the gradient projection method using a nonmonotone line search.

NOTES AND REFERENCES

SQP methods were first proposed in 1963 by Wilson [306] and were developed in the 1970s by Garcia-Palomares and Mangasarian [117], Han [163, 164], and Powell [247, 250, 249], among others. Trust-region variants are studied by Vardi [295], Celis, Dennis, and Tapia [56], and Byrd, Schnabel, and Shultz [55]. See Boggs and Tolle [33] and Gould, Orban, and Toint [147] for literature surveys.


The SLQP approach was proposed by Fletcher and Sainz de la Maza [108] and was further developed by Chin and Fletcher [59] and Byrd et al. [49]. The latter paper discusses how to update the LP trust region and many other details of implementation. The technique for updating the penalty parameter implemented in Algorithm 18.5 is discussed in [49, 47]. The $S\ell_1$ QP method was proposed by Fletcher; see [101] for a complete discussion of this method.


Some analysis shows that several—but not all—of the good properties of BFGS updating are preserved by damped BFGS updating. Numerical experiments exposing the weakness of the approach are reported by Powell [254]. Second-order correction strategies were proposed by Coleman and Conn [65], Fletcher [100], Gabay [116], and Mayne and Polak [204]. The watchdog technique was proposed by Chamberlain et al. [57] and other nonmonotone strategies are described by Bonnans et al. [36]. For a comprehensive discussion of second-order correction and nonmonotone techniques, see the book by Conn, Gould, and Toint [74].


Two filter SQP algorithms are described by Fletcher and Leyffer [105] and Fletcher, Leyffer, and Toint [106]. It is not yet known whether the filter strategy has advantages over merit functions. Both approaches are undergoing development and improved implementations can be expected in the future. Theorem 18.3 is proved by Powell [252] and Theorem 18.5 by Boggs, Tolle, and Wang [34].



EXERCISES

 **18.1** Show that in the quadratic program (18.7) we can replace the linear term $\nabla f_k^T p$ by $\nabla_x \mathcal{L}(x_k, \lambda_k)^T p$ without changing the solution.

 **18.2** Prove Theorem 18.4.

 **18.3** Write a program that implements Algorithm 18.1. Use it to solve the problem


$$\min e^{x_1 x_2 x_3 x_4 x_5} - \frac{1}{2}(x_1^3 + x_2^3 + 1)^2 \quad (18.66)$$


$$\text{subject to } x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 - 10 = 0, \quad (18.67)$$


$$x_2 x_3 - 5 x_4 x_5 = 0, \quad (18.68)$$


$$x_1^3 + x_2^3 + 1 = 0. \quad (18.69)$$


Use the starting point $x_0 = (-1.71, 1.59, 1.82, -0.763, -0.763)^T$. The solution is $x^* = (-1.8, 1.7, 1.9, -0.8, -0.8)^T$.


 **18.4** Show that the damped BFGS updating satisfies (18.17).

 **18.5** Consider the constraint $x_1^2 + x_2^2 = 1$. Write the linearized constraints (18.7b) at the following points: $(0, 0)^T$, $(0, 1)^T$, $(0.1, 0.02)^T$, $-(0.1, 0.02)^T$.

 **18.6** Prove Theorem 18.2 for the case in which the merit function is given by $\phi(x; \mu) = f(x) + \mu \|c(x)\|_q$, where $q > 0$. Use this lemma to show that the condition that ensures descent is given by $\mu > \|\lambda_{k+1}\|_r$, where $r > 0$ satisfies $r^{-1} + q^{-1} = 1$.

 **18.7** Write a program that implements the reduced-Hessian method given by (18.18), (18.19a), (18.21), (18.23). Use your program to solve the problem given in Exercise 18.3.

 **18.8** Show that the constraints (18.50b)–(18.50e) are always consistent.

 **18.9** Show that the feasibility problem (18.45a)–(18.45b) always has a solution v_k lying in the range space of A_k^T . Hint: First show that if the trust-region constraint (18.45b) is active, v_k lies in the range space of A_k^T . Next, show that if the trust region is inactive, the minimum-norm solution of (18.45a) lies in the range space of A_k^T .