

CHAPTER *15*

Fundamentals of Algorithms for Nonlinear Constrained Optimization

In this chapter, we begin our discussion of algorithms for solving the general constrained optimization problem

$$\min_{x \in \mathbb{R}^n} f(x) \quad \text{subject to} \quad \begin{aligned} c_i(x) &= 0, & i \in \mathcal{E}, \\ c_i(x) &\geq 0, & i \in \mathcal{I}, \end{aligned} \quad (15.1)$$

where the objective function f and the constraint functions c_i are all smooth, real-valued functions on a subset of \mathbb{R}^n , and \mathcal{I} and \mathcal{E} are finite index sets of inequality and equality constraints, respectively. In Chapter 12, we used this general statement of the problem

to derive optimality conditions that characterize its solutions. This theory is useful for motivating the various algorithms discussed in the remainder of the book, which differ from each other in fundamental ways but are all iterative in nature. They generate a sequence of estimates of the solution x^* that, we hope, tend toward a solution. In some cases, they also generate a sequence of guesses for the Lagrange multipliers associated with the constraints. As in the chapters on unconstrained optimization, we study only algorithms for finding local solutions of (15.1); the problem of finding a global solution is outside the scope of this book.

We note that this chapter is not concerned with individual algorithms themselves, but rather with fundamental concepts and building blocks that are common to more than one algorithm. After reading Sections 15.1 and 15.2, the reader may wish to glance at the material in Sections 15.3, 15.4, 15.5, and 15.6, and return to these sections as needed during study of subsequent chapters.

15.1 CATEGORIZING OPTIMIZATION ALGORITHMS

We now catalog the algorithmic approaches presented in the rest of the book. No standard taxonomy exists for nonlinear optimization algorithms; in the remaining chapters we have grouped the various approaches as follows.

I. In Chapter 16 we study algorithms for solving *quadratic programming* problems. We consider this category separately because of its intrinsic importance, because its particular characteristics can be exploited by efficient algorithms, and because quadratic programming subproblems need to be solved by sequential quadratic programming methods and certain interior-point methods for nonlinear programming. We discuss active set, interior-point, and gradient projection methods.

II. In Chapter 17 we discuss *penalty* and *augmented Lagrangian* methods. By combining the objective function and constraints into a *penalty function*, we can attack problem (15.1) by solving a sequence of unconstrained problems. For example, if only equality constraints are present in (15.1), we can define the quadratic penalty function as

$$f(x) + \frac{\mu}{2} \sum_{i \in \mathcal{E}} c_i^2(x), \quad (15.2)$$

where $\mu > 0$ is referred to as a *penalty parameter*. We minimize this unconstrained function, for a series of increasing values of μ , until the solution of the constrained optimization problem is identified to sufficient accuracy.

If we use an *exact* penalty function, it may be possible to find a local solution of (15.1) by solving a single unconstrained optimization problem. For the equality-constrained problem,

the function defined by

$$f(x) + \mu \sum_{i \in \mathcal{E}} |c_i(x)|,$$

is usually an exact penalty function, for a sufficiently large value of $\mu > 0$. Although they often are nondifferentiable, exact penalty functions can be minimized by solving a sequence of smooth subproblems.

In *augmented Lagrangian methods*, we define a function that combines the properties of the Lagrangian function (12.33) and the quadratic penalty function (15.2). This so-called augmented Lagrangian function has the following form for equality-constrained problems:

$$\mathcal{L}_A(x, \lambda; \mu) = f(x) - \sum_{i \in \mathcal{E}} \lambda_i c_i(x) + \frac{\mu}{2} \sum_{i \in \mathcal{E}} c_i^2(x).$$

Methods based on this function fix λ to some estimate of the optimal Lagrange multiplier vector and fix μ to some positive value, then find a value of x that approximately minimizes $\mathcal{L}_A(\cdot, \lambda; \mu)$. At this new x -iterate, λ and μ may be updated; then the process is repeated. This approach avoids certain drawbacks associated with the minimization of the quadratic penalty function (15.2).

III. In Chapter 18 we describe *sequential quadratic programming* (SQP) methods, which model (15.1) by a quadratic programming subproblem at each iterate and define the search direction to be the solution of this subproblem. In the basic SQP method, we define the search direction p_k at the iterate (x_k, λ_k) to be the solution of

$$\min_p \quad \frac{1}{2} p^T \nabla_{xx}^2 \mathcal{L}(x_k, \lambda_k) p + \nabla f(x_k)^T p \quad (15.3a)$$

$$\text{subject to} \quad \nabla c_i(x_k)^T p + c_i(x_k) = 0, \quad i \in \mathcal{E}, \quad (15.3b)$$

$$\nabla c_i(x_k)^T p + c_i(x_k) \geq 0, \quad i \in \mathcal{I}, \quad (15.3c)$$

where \mathcal{L} is the Lagrangian function defined in (12.33). The objective in this subproblem is an approximation to the change in the Lagrangian function in moving from x_k to $x_k + p$, while the constraints are linearizations of the constraints in (15.1). A trust-region constraint may be added to (15.3) to control the length and quality of the step, and quasi-Newton approximate Hessians can be used in place of $\nabla_{xx}^2 \mathcal{L}(x_k, \lambda_k)$. In a variant called *sequential linear-quadratic programming*, the step p_k is computed in two stages. First, we solve a linear program that is defined by omitting the first (quadratic) term from the objective (15.3a) and adding a trust-region constraint to (15.3). Next, we obtain the step p_k by solving an equality-constrained subproblem in which the constraints active at the solution of the linear program are imposed as equalities, while all other constraints are ignored.

IV. In Chapter 19 we study *interior-point methods for nonlinear programming*. These methods can be viewed as extensions of the primal-dual interior-point methods for linear

programming discussed in Chapter 14. We can also view them as *barrier methods* that generate steps by solving the problem

$$\min_{x,s} f(x) - \mu \sum_{i=1}^m \log s_i \quad (15.4a)$$

$$\text{subject to } c_i(x) = 0, \quad i \in \mathcal{E}, \quad (15.4b)$$

$$c_i(x) - s_i = 0, \quad i \in \mathcal{I}, \quad (15.4c)$$

for some positive value of the barrier parameter μ , where the variables $s_i > 0$ are slacks. Interior-point methods constitute the newest class of methods for nonlinear programming and have already proved to be formidable competitors of sequential quadratic programming methods.

The algorithms in categories I, III, and IV make use of elimination techniques, in which the constraints are used to eliminate some of the degrees of freedom in the problem. As a background to those algorithms, we discuss elimination in Section 15.3. In later sections we discuss merit functions and filters, which are important mechanisms for promoting convergence of nonlinear programming algorithms from remote starting points.

15.2 THE COMBINATORIAL DIFFICULTY OF INEQUALITY-CONSTRAINED PROBLEMS

One of the main challenges in solving nonlinear programming problems lies in dealing with inequality constraints—in particular, in deciding which of these constraints are active at the solution and which are not. One approach, which is the essence of active-set methods, starts by making a guess of the optimal active set \mathcal{A}^* , that is, the set of constraints that are satisfied as equalities at a solution. We call our guess the *working set* and denote it by \mathcal{W} . We then solve a problem in which the constraints in the working set are imposed as equalities and the constraints not in \mathcal{W} are ignored. We then check to see if there is a choice of Lagrange multipliers such that the solution x^* obtained for this \mathcal{W} satisfies the KKT conditions (12.34). If so, we accept x^* as a local solution of (15.1). Otherwise, we make a different choice of \mathcal{W} and repeat the process. This approach is based on the observation that, in general, it is much simpler to solve equality-constrained problems than to solve nonlinear programs.

The number of choices for working set \mathcal{W} may be very large—up to $2^{|\mathcal{I}|}$, where $|\mathcal{I}|$ is the number of inequality constraints. We arrive at this estimate by observing that we can make one of two choices for each $i \in \mathcal{I}$: to include it in \mathcal{W} or leave it out. Since the number of possible working sets grows exponentially with the number of inequalities—a phenomenon which we refer to as the *combinatorial difficulty* of nonlinear programming—we cannot hope to design a practical algorithm by considering all possible choices for \mathcal{W} .

The following example suggests that even for a small number of inequality constraints, determination of the optimal active set is not a simple task.

□ **EXAMPLE 15.1**

Consider the problem

$$\begin{aligned} \min_{x,y} f(x,y) &\stackrel{\text{def}}{=} \frac{1}{2}(x-2)^2 + \frac{1}{2}(y-\frac{1}{2})^2 & (15.5) \\ &(x+1)^{-1} - y - \frac{1}{4} \geq 0, \\ \text{subject to} & \quad x \geq 0, \\ & \quad y \geq 0. \end{aligned}$$

We label the constraints, in order, with the indices 1 through 3. Figure 15.1 illustrates the contours of the objective function (dashed circles). The feasible region is the region enclosed by the curve and the two axes. We see that only the first constraint is active at the solution, which is $(x^*, y^*)^T = (1.953, 0.089)^T$.

Let us now apply the working-set approach described above to (15.5), considering all $2^3 = 8$ possible choices of \mathcal{W} .

We consider first the possibility that no constraints are active at the solution, that is, $\mathcal{W} = \emptyset$. Since $\nabla f = (x-2, y-1/2)^T$, we see that the unconstrained minimum of f lies outside the feasible region. Hence, the optimal active set cannot be empty.

There are seven further possibilities. First, all three constraints could be active (that is, $\mathcal{W} = \{1, 2, 3\}$). A glance at Figure 15.1 shows that this does not happen for our problem; the three constraints do not share a common point of intersection. Three further possibilities are obtained by making a single constraint active (that is, $\mathcal{W} = \{1\}$, $\mathcal{W} = \{2\}$, and $\mathcal{W} = \{3\}$),

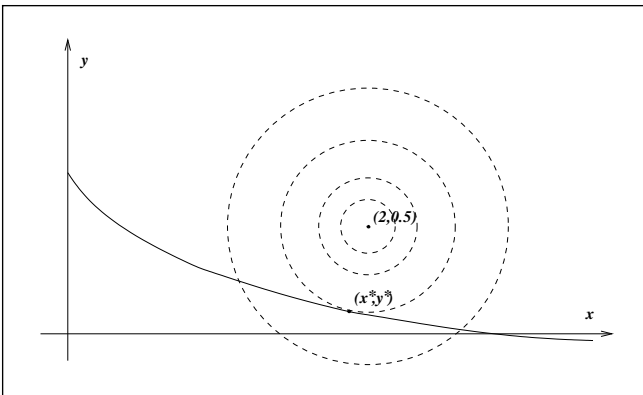


Figure 15.1 Graphical illustration of problem (15.5).

while the final three possibilities are obtained by making exactly two constraints active (that is, $\mathcal{W} = \{1, 2\}$, $\mathcal{W} = \{1, 3\}$, and $\mathcal{W} = \{2, 3\}$). We consider three of these cases in detail.

- $\mathcal{W} = \{2\}$; that is, only the constraint $x = 0$ is active. If we minimize f enforcing only this constraint, we obtain the point $(0, 1/2)^T$. A check of the KKT conditions (12.34) shows that no matter how we choose the Lagrange multipliers, we cannot satisfy all these conditions at $(0, 1/2)^T$. (We must have $\lambda_1 = \lambda_3 = 0$ to satisfy (12.34e), which implies that we must set $\lambda_2 = -2$ to satisfy (12.34a); but this value of λ_2 violates the condition (12.34d).)
- $\mathcal{W} = \{1, 3\}$, which yields the single feasible point $(3, 0)^T$. Since constraint 2 is inactive at this point, we have $\lambda_2 = 0$, so by solving (12.34a) for the other Lagrange multipliers, we obtain $\lambda_1 = -16$ and $\lambda_3 = -16.5$. These values are negative, so they violate (12.34d), and $x = (3, 0)^T$ cannot be a solution of (15.1).
- $\mathcal{W} = \{1\}$. Solving the equality-constrained problem in which the first constraint is active, we obtain $(x, y)^T = (1.953, 0.089)^T$ with Lagrange multiplier $\lambda_1 = 0.411$. It is easy to see that by setting $\lambda_2 = \lambda_3 = 0$, the remaining KKT conditions (12.34) are satisfied, so we conclude that this is a KKT point. Furthermore, it is easy to show that the second-order sufficient conditions are satisfied, as the Hessian of the Lagrangian is positive definite. □

Even for this small example, we see that it is exhausting to consider all possible choices for \mathcal{W} . Figure 15.1 suggests, however, that some choices of \mathcal{W} can be eliminated from consideration if we make use of knowledge of the functions that define the problem, and their derivatives. In fact, the active set methods described in Chapter 16 use this kind of information to make a series of educated guesses for the working set, avoiding choices of \mathcal{W} that obviously will not lead to a solution of (15.1).

A different approach is followed by interior-point (or barrier) methods discussed in Chapter 19. These methods generate iterates that stay away from the boundary of the feasible region defined by the inequality constraints. As the solution of the nonlinear program is approached, the barrier effects are weakened to permit an increasingly accurate estimate of the solution. In this manner, interior-point methods avoid the combinatorial difficulty of nonlinear programming.

15.3 ELIMINATION OF VARIABLES

When dealing with constrained optimization problems, it is natural to try to use the constraints to eliminate some of the variables from the problem, to obtain a simpler problem with fewer degrees of freedom. Elimination techniques must be used with care, however, as they may alter the problem or introduce ill conditioning.

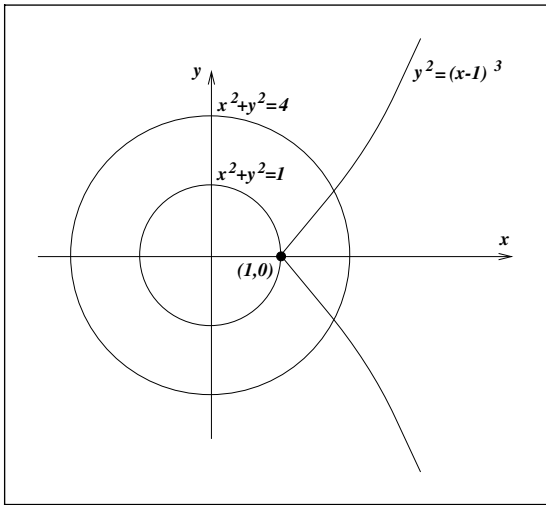


Figure 15.2
The danger of nonlinear
elimination.

We begin with an example in which it is safe and convenient to eliminate variables. In the problem

$$\min f(x) = f(x_1, x_2, x_3, x_4) \quad \text{subject to} \quad \begin{aligned} x_1 + x_3^2 - x_4x_3 &= 0, \\ -x_2 + x_4 + x_3^2 &= 0, \end{aligned}$$

there is no risk in setting

$$x_1 = x_4x_3 - x_3^2, \quad x_2 = x_4 + x_3^2,$$

to obtain a function of two variables

$$h(x_3, x_4) = f(x_4x_3 - x_3^2, x_4 + x_3^2, x_3, x_4),$$

which we can minimize using the unconstrained optimization techniques described in earlier chapters.

The dangers of nonlinear elimination are illustrated in the following example.

□ **EXAMPLE 15.2** (FLETCHER [101])

Consider the problem

$$\min x^2 + y^2 \quad \text{subject to} \quad (x - 1)^3 = y^2.$$

The contours of the objective function and the constraints are illustrated in Figure 15.2, which shows that the solution is $(x, y) = (1, 0)$.

We attempt to solve this problem by eliminating y . By doing so, we obtain

$$h(x) = x^2 + (x - 1)^3.$$

Clearly, $h(x) \rightarrow -\infty$ as $x \rightarrow -\infty$. By blindly applying this transformation we may conclude that the problem is unbounded, but this view ignores the fact that the constraint $(x - 1)^3 = y^2$ implicitly imposes the bound $x \geq 1$ that is active at the solution. Hence, if we wish to eliminate y , we should explicitly introduce the bound $x \geq 1$ into the problem. □

This example shows that the use of nonlinear equations to eliminate variables may result in errors that can be difficult to trace. For this reason, nonlinear elimination is not used by most optimization algorithms. Instead, many algorithms linearize the constraints and apply elimination techniques to the simplified problem. We now describe systematic procedures for performing variable elimination using linear constraints.

SIMPLE ELIMINATION USING LINEAR CONSTRAINTS

We consider the minimization of a nonlinear function subject to a set of linear equality constraints,

$$\min f(x) \quad \text{subject to } Ax = b, \quad (15.6)$$

where A is an $m \times n$ matrix with $m \leq n$. Suppose for simplicity that A has full row rank. (If such is not the case, we find either that the problem is inconsistent or that some of the constraints are redundant and can be deleted without affecting the solution of the problem.) Under this assumption, we can find a subset of m columns of A that is linearly independent. If we gather these columns into an $m \times m$ matrix B and define an $n \times n$ permutation matrix P that swaps these columns to the first m column positions in A , we can write

$$AP = [B \mid N], \quad (15.7)$$

where N denotes the $n - m$ remaining columns of A . (The notation here is consistent with that of Chapter 13, where we discussed similar concepts in the context of linear programming.) We define the subvectors $x_B \in \mathbb{R}^m$ and $x_N \in \mathbb{R}^{n-m}$ as follows:

$$\begin{bmatrix} x_B \\ x_N \end{bmatrix} = P^T x, \quad (15.8)$$

and call x_B the *basic variables* and B the *basis matrix*. Noting that $PP^T = I$, we can rewrite the constraint $Ax = b$ as

$$b = Ax = AP(P^T x) = Bx_B + Nx_N.$$

By rearranging this formula, we deduce that the basic variables can be expressed as follows:

$$x_B = B^{-1}b - B^{-1}Nx_N. \quad (15.9)$$

We can therefore compute a feasible point for the constraints $Ax = b$ by choosing *any* value of x_N and then setting x_B according to the formula (15.9). The problem (15.6) is therefore equivalent to the unconstrained problem

$$\min_{x_N} h(x_N) \stackrel{\text{def}}{=} f \left(P \begin{bmatrix} B^{-1}b - B^{-1}Nx_N \\ x_N \end{bmatrix} \right). \quad (15.10)$$

We refer to the substitution in (15.9) as *simple elimination of variables*.

This discussion shows that a nonlinear optimization problem with linear equality constraints is, from a mathematical point of view, the same as an unconstrained problem.

□ EXAMPLE 15.3

Consider the problem

$$\min \sin(x_1 + x_2) + x_3^2 + \frac{1}{3}(x_4 + x_5^4 + x_6/2) \quad (15.11a)$$

$$\begin{aligned} \text{subject to} \quad & 8x_1 - 6x_2 + x_3 + 9x_4 + 4x_5 = 6 \\ & 3x_1 + 2x_2 - x_4 + 6x_5 + 4x_6 = -4. \end{aligned} \quad (15.11b)$$

By defining the permutation matrix P so as to reorder the components of x as $x^T = (x_3, x_6, x_1, x_2, x_4, x_5)^T$, we find that the coefficient matrix AP is

$$AP = \left[\begin{array}{cc|cccc} 1 & 0 & 8 & -6 & 9 & 4 \\ 0 & 4 & 3 & 2 & -1 & 6 \end{array} \right].$$

The basis matrix B is diagonal and therefore easy to invert. We obtain from (15.9) that

$$\begin{bmatrix} x_3 \\ x_6 \end{bmatrix} = - \begin{bmatrix} 8 & -6 & 9 & 4 \\ 3 & 1 & -1 & 3 \\ 4 & 2 & -1 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_4 \\ x_5 \end{bmatrix} + \begin{bmatrix} 6 \\ -1 \end{bmatrix}. \quad (15.12)$$

By substituting for x_3 and x_6 in (15.11a), the problem becomes

$$\begin{aligned} \min_{x_1, x_2, x_4, x_5} \quad & \sin(x_1 + x_2) + (8x_1 - 6x_2 + 9x_4 + 4x_5 - 6)^2 \\ & + \frac{1}{3}(x_4 + x_5^4 - [(1/2) + (3/8)x_1 + (1/4)x_2 - (1/8)x_4 + (3/4)x_5]). \end{aligned} \quad (15.13)$$

We could have chosen two other columns of the coefficient matrix A (that is, two variables other than x_3 and x_6) as the basis for elimination in the system (15.11b), but the matrix $B^{-1}N$ would not have been so simple. □

A set of m independent columns can be selected, in general, by means of Gaussian elimination. In the parlance of linear algebra, we can compute the row echelon form of the matrix and choose the pivot columns as the columns of the basis B . Ideally, we would like B to be easy to factor and well conditioned. A technique that suits these purposes is a sparse Gaussian elimination approach that attempts to preserve sparsity while keeping rounding errors under control. A well-known implementation of this algorithm is MA48 from the HSL library [96]. As we discuss below, however, there is no guarantee that the Gaussian elimination process will identify the best choice of basis matrix.

There is an interesting interpretation of the simple elimination-of-variables approach that we have just described. To simplify the notation, we will assume from now on that the coefficient matrix is already given to us so that the basic columns appear in the first m positions, that is, $P = I$.

From (15.8) and (15.9) we see that any feasible point x for the linear constraints in (15.6) can be written as

$$\begin{bmatrix} x_B \\ x_N \end{bmatrix} = x = Yb + ZX_N, \quad (15.14)$$

where

$$Y = \begin{bmatrix} B^{-1} \\ 0 \end{bmatrix}, \quad Z = \begin{bmatrix} -B^{-1}N \\ I \end{bmatrix}. \quad (15.15)$$

Note that Z has $n - m$ linearly independent columns (because of the presence of the identity matrix in the lower block) and that it satisfies $AZ = 0$. Therefore, Z is a *basis for the null space* of A . In addition, the columns of Y and the columns of Z form a linearly independent set. We note also from (15.15), (15.7) that Yb is a particular solution of the linear constraints $Ax = b$.

In other words, the simple elimination technique expresses feasible points as the sum of a particular solution of $Ax = b$ (the first term in (15.14)) plus a displacement along the

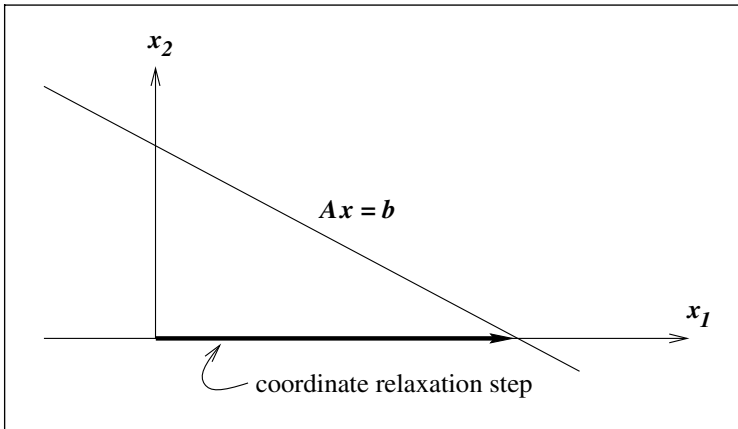


Figure 15.3 Simple elimination, showing the coordinate relaxation step obtained by choosing the basis to be the first column of A .

null space of the constraints (the second term in (15.14)). The relations (15.14), (15.15) indicate that the particular Yb solution is obtained by holding $n - m$ components of x at zero while relaxing the other m components (the ones in x_b) until they reach the constraints. The particular solution Yb is sometimes known as the coordinate relaxation step. In Figure 15.3, we see the coordinate relaxation step Yb obtained by choosing the basis matrix B to be the first column of A . If we were to choose B to be the second column of A , the coordinate relaxation step would lie along the x_2 axis.

Simple elimination is inexpensive but can give rise to numerical instabilities. If the feasible set in Figure 15.3 consisted of a line that was almost parallel to the x_1 axis, the coordinate relaxation along this axis would be very large in magnitude. We would then be computing x as the difference of very large vectors, giving rise to numerical cancellation. In that situation it would be preferable to choose a particular solution along the x_2 axis, that is, to select a different basis. Selection of the best basis is, therefore, not a straightforward task in general. To overcome the dangers of an excessively large coordinate relaxation step, we could define the particular solution Yb as the minimum-norm step to the constraints. This approach is a special case of more general elimination strategies, which we now describe.

GENERAL REDUCTION STRATEGIES FOR LINEAR CONSTRAINTS

To generalize (15.14) and (15.15), we choose matrices $Y \in \mathbb{R}^{n \times m}$ and $Z \in \mathbb{R}^{n \times (n-m)}$ with the following properties:

$$[Y \mid Z] \in \mathbb{R}^{n \times n} \text{ is nonsingular, } AZ = 0. \quad (15.16)$$

These properties indicate that, as in (15.15), the columns of Z are a basis for the null space of A . Since A has full row rank, so does $A[Y \mid Z] = [AY \mid 0]$, so it follows that the $m \times m$ matrix AY is nonsingular. We now express any solution of the linear constraints $Ax = b$ as

$$x = Yx_Y + Zx_Z, \quad (15.17)$$

for some vectors $x_Y \in \mathbb{R}^m$ and $x_Z \in \mathbb{R}^{n-m}$. By substituting (15.17) into the constraints $Ax = b$, we obtain

$$Ax = (AY)x_Y = b;$$

hence by nonsingularity of AY , x_Y can be written explicitly as

$$x_Y = (AY)^{-1}b. \quad (15.18)$$

By substituting this expression into (15.17), we conclude that any vector x of the form

$$x = Y(AY)^{-1}b + Zx_Z \quad (15.19)$$

satisfies the constraints $Ax = b$ for any choice of $x_Z \in \mathbb{R}^{n-m}$. Therefore, the problem (15.6) can be restated equivalently as the following unconstrained problem

$$\min_{x_Z} f(Y(AY)^{-1}b + Zx_Z). \quad (15.20)$$

Ideally, we would like to choose Y in such a way that the matrix AY is as well conditioned as possible, since it needs to be factorized to give the particular solution $Y(AY)^{-1}b$. We can do this by computing Y and Z by means of a QR factorization of A^T , which has the form

$$A^T \Pi = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} R \\ 0 \end{bmatrix}, \quad (15.21)$$

where $\begin{bmatrix} Q_1 & Q_2 \end{bmatrix}$ is orthogonal. The submatrices Q_1 and Q_2 have orthonormal columns and are of dimension $n \times m$ and $n \times (n - m)$, while R is $m \times m$ upper triangular and nonsingular and Π is an $m \times m$ permutation matrix. (See the discussion following (A.24) in the Appendix for further details.) We now define

$$Y = Q_1, \quad Z = Q_2, \quad (15.22)$$

so that the columns of Y and Z form an *orthonormal basis* of \mathbb{R}^n . If we expand (15.21) and do a little rearrangement, we obtain

$$AY = \Pi R^T, \quad AZ = 0.$$

Therefore, Y and Z have the desired properties, and the condition number of AY is the same as that of R , which in turn is the same as that of A itself. From (15.19) we see that any solution of $Ax = b$ can be expressed as

$$x = Q_1 R^{-T} \Pi^T b + Q_2 x_z,$$

for some vector x_z . The computation $R^{-T} \Pi^T b$ can be carried out inexpensively, at the cost of a single triangular substitution.

A simple computation shows that the particular solution $Q_1 R^{-T} \Pi^T b$ can also be written as $A^T (AA^T)^{-1} b$. This vector is the solution of the following problem:

$$\min \|x\|^2 \quad \text{subject to } Ax = b;$$

that is, it is the minimum-norm solution of $Ax = b$. See Figure 15.5 for an illustration of this step.

Elimination via the orthogonal basis (15.22) is ideal from the point of view of numerical stability. The main cost associated with this reduction strategy is in computing the QR factorization (15.21). Unfortunately, for problems in which A is large and sparse, a sparse QR factorization can be much more costly to compute than the sparse Gaussian elimination strategy used in simple elimination. Therefore, other elimination strategies have been developed that seek a compromise between these two techniques; see Exercise 15.7.

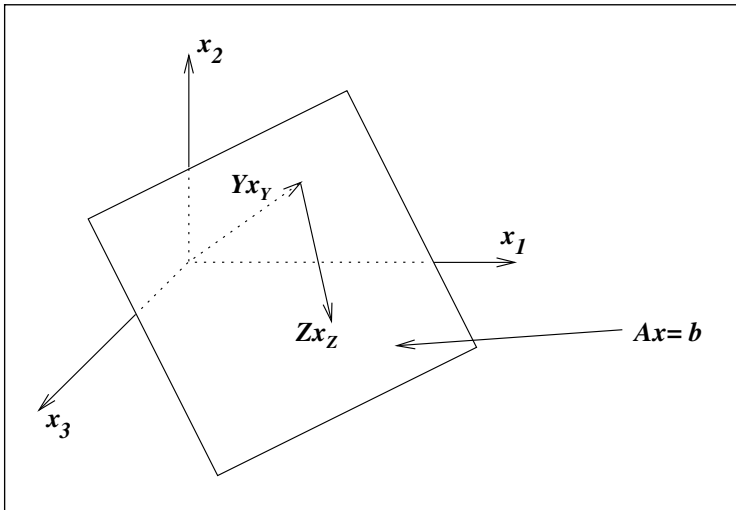


Figure 15.4 General elimination: Case in which $A \in \mathbb{R}^{1 \times 3}$, showing the particular solution and a step in the null space of A .

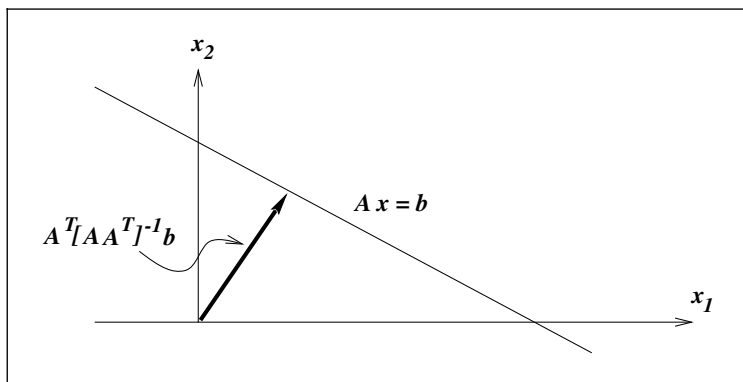


Figure 15.5 The minimum-norm step.

EFFECT OF INEQUALITY CONSTRAINTS

Elimination of variables is not always beneficial if inequality constraints are present alongside the equalities. For instance, if problem (15.11) had the additional constraint $x \geq 0$, then after eliminating the variables x_3 and x_6 , we would be left with the problem of minimizing the function in (15.13) subject to the constraints

$$\begin{aligned} (x_1, x_2, x_4, x_5) &\geq 0, \\ 8x_1 - 6x_2 + 9x_4 + 4x_5 &\leq 6, \\ (3/4)x_1 + (1/2)x_2 - (1/4)x_4 + (3/2)x_5 &\leq -1. \end{aligned}$$

Hence, the cost of eliminating the equality constraints (15.11b) is to make the inequalities more complicated than the simple bounds $x \geq 0$. For many algorithms, this transformation will not yield any benefit.

If, however, problem (15.11) included the general inequality constraint $3x_1 + 2x_3 \geq 1$, the elimination (15.12) would transform the problem into one of minimizing the function in (15.13) subject to the inequality constraint

$$-13x_1 + 12x_2 - 18x_4 - 8x_5 \geq -11. \quad (15.23)$$

In this case, the inequality constraint would not become much more complicated after elimination of the equality constraints, so it is probably worthwhile to perform the elimination.

15.4 MERIT FUNCTIONS AND FILTERS

Suppose that an algorithm for solving the nonlinear programming problem (15.1) generates a step that reduces the objective function but increases the violation of the constraints. Should we accept this step?

This question is not easy to answer. We must look for a way to balance the twin (often competing) goals of reducing the objective function and satisfying the constraints. Merit functions and filters are two approaches for achieving this balance. In a typical constrained optimization algorithm, a step p will be accepted only if it leads to a sufficient reduction in the merit function ϕ or if it is acceptable to the filter. These concepts are explained in the rest of the section.

MERIT FUNCTIONS

In unconstrained optimization, the objective function f is the natural choice for the merit function. All the unconstrained optimization methods described in this book require that f be decreased at each step (or at least within a certain number of iterations). In feasible methods for constrained optimization in which the starting point and all subsequent iterates satisfy all the constraints in the problem, the objective function is still an appropriate merit function. On the other hand, algorithms that allow iterates to violate the constraints require some means to assess the quality of the steps and iterates. The merit function in this case combines the objective with measures of constraint violation.

A popular choice of merit function for the nonlinear programming problem (15.1) is the ℓ_1 penalty function defined by

$$\phi_1(x; \mu) = f(x) + \mu \sum_{i \in \mathcal{E}} |c_i(x)| + \mu \sum_{i \in \mathcal{I}} [c_i(x)]^-, \quad (15.24)$$

where we use the notation $[z]^- = \max\{0, -z\}$. The positive scalar μ is the *penalty parameter*, which determines the weight that we assign to constraint satisfaction relative to minimization of the objective. The ℓ_1 merit function ϕ_1 is not differentiable because of the presence of the absolute value and $[\cdot]^-$ functions, but it has the important property of being *exact*.

Definition 15.1 (Exact Merit Function).

A merit function $\phi(x; \mu)$ is exact if there is a positive scalar μ^ such that for any $\mu > \mu^*$, any local solution of the nonlinear programming problem (15.1) is a local minimizer of $\phi(x; \mu)$.*

We show in Theorem 17.3 that, under certain assumptions, the ℓ_1 merit function $\phi_1(x; \mu)$ is exact and that the threshold value μ^* is given by

$$\mu^* = \max\{|\lambda_i^*|, i \in \mathcal{E} \cup \mathcal{I}\},$$

where the λ_i^* denote the Lagrange multipliers associated with an optimal solution x^* . Since the optimal Lagrange multipliers are, however, not known in advance, algorithms based on the ℓ_1 merit function contain rules for adjusting the penalty parameter whenever there is reason to believe that it is not large enough (or is excessively large). These rules depend on the choice of optimization algorithm and are discussed in the next chapters.

Another useful merit function is the exact ℓ_2 function, which for equality-constrained problems takes the form

$$\phi_2(x; \mu) = f(x) + \mu \|c(x)\|_2. \quad (15.25)$$

This function is nondifferentiable because the 2-norm term is not squared; its derivative is not defined at x for which $c(x) = 0$.

Some merit functions are both smooth and exact. To ensure that both properties hold, we must include additional terms in the merit function. For equality-constrained problems, *Fletcher's augmented Lagrangian* is given by

$$\phi_F(x; \mu) = f(x) - \lambda(x)^T c(x) + \frac{1}{2} \mu \sum_{i \in \mathcal{E}} c_i(x)^2, \quad (15.26)$$

where $\mu > 0$ is the penalty parameter and

$$\lambda(x) = [A(x)A(x)^T]^{-1} A(x) \nabla f(x). \quad (15.27)$$

(Here $A(x)$ denotes the Jacobian of $c(x)$.) Although this merit function has some interesting theoretical properties, it has practical limitations, including the expense of solving for $\lambda(x)$ in (15.27).

A quite different merit function is the (standard) augmented Lagrangian in x and λ , which for equality-constrained problems has the form

$$\mathcal{L}_A(x, \lambda; \mu) = f(x) - \lambda^T c(x) + \frac{1}{2} \mu \|c(x)\|_2^2. \quad (15.28)$$

We assess the acceptability of a trial point (x^+, λ^+) by comparing the value of $\mathcal{L}_A(x^+, \lambda^+; \mu)$ with the value at the current iterate, (x, λ) . Strictly speaking, \mathcal{L}_A is not a merit function in the sense that a solution (x^*, λ^*) of the nonlinear programming problem is not in general a minimizer of $\mathcal{L}_A(x, \lambda; \mu)$ but only a stationary point. Although some sequential quadratic programming methods use \mathcal{L}_A successfully as a merit function by adaptively modifying μ and λ , we will not consider its use as a merit function further. Instead, we will focus primarily on the nonsmooth exact penalty functions ϕ_1 and ϕ_2 .

A trial step $x^+ = x + \alpha p$ generated by a line search algorithm will be accepted if it produces a *sufficient decrease* in the merit function $\phi(x; \mu)$. One way to define this concept is analogous to the condition (3.4) used in unconstrained optimization, where the amount

of decrease is not too small relative to the predicted change in the function over the step. The ℓ_1 and ℓ_2 merit functions are not differentiable, but they have a directional derivative. (See (A.51) for background on directional derivatives.) We write the directional derivative of $\phi(x; \mu)$ in the direction p as

$$D(\phi(x; \mu); p).$$

In a line search method, the sufficient decrease condition requires the steplength parameter $\alpha > 0$ to be small enough that the inequality

$$\phi(x + \alpha p; \mu) \leq \phi(x; \mu) + \eta \alpha D(\phi(x; \mu); p), \quad (15.29)$$

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

Trust-region methods typically use a quadratic model $q(p)$ to estimate the value of the merit function ϕ after a step p ; see Section 18.5. The sufficient decrease condition can be stated in terms of a decrease in this model, as follows

$$\phi(x + p; \mu) \leq \phi(x; \mu) - \eta(q(0) - q(p)), \quad (15.30)$$

for some $\eta \in (0, 1)$. (The final term in (15.30) is positive, because the step p is computed to decrease the model q .)

FILTERS

Filter techniques are step acceptance mechanisms based on ideas from multiobjective optimization. Our derivation starts with the observation that nonlinear programming has two goals: minimization of the objective function and the satisfaction of the constraints. If we define a measure of infeasibility as

$$h(x) = \sum_{i \in \mathcal{E}} |c_i(x)| + \sum_{i \in \mathcal{I}} [c_i(x)]^-, \quad (15.31)$$

we can write these two goals as

$$\min_x f(x) \quad \text{and} \quad \min_x h(x). \quad (15.32)$$

Unlike merit functions, which combine both problems into a single minimization problem, filter methods keep the two goals in (15.32) separate. Filter methods accept a trial step x^+ as a new iterate if the pair $(f(x^+), h(x^+))$ is not *dominated* by a previous pair $(f_l, h_l) = (f(x_l), h(x_l))$ generated by the algorithm. These concepts are defined as follows.

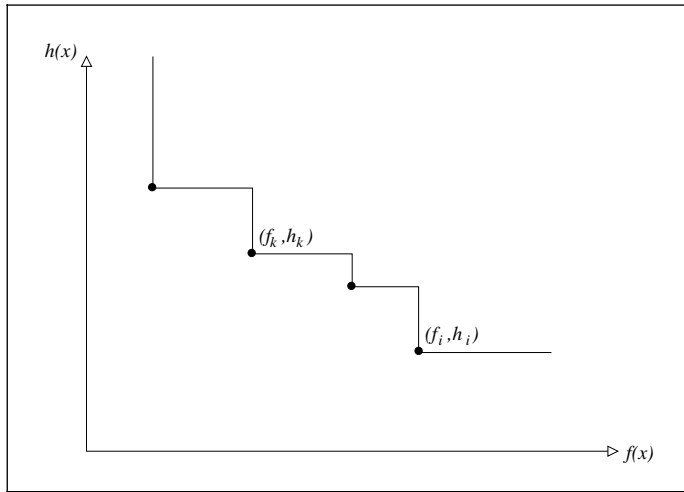


Figure 15.6 Graphical illustration of a filter with four pairs.

Definition 15.2.

- (a) A pair (f_k, h_k) is said to dominate another pair (f_l, h_l) if both $f_k \leq f_l$ and $h_k \leq h_l$.
- (b) A filter is a list of pairs (f_l, h_l) such that no pair dominates any other.
- (c) An iterate x_k is said to be acceptable to the filter if (f_k, h_k) is not dominated by any pair in the filter.

When an iterate x_k is acceptable to the filter, we (normally) add (f_k, h_k) to the filter and remove any pairs that are dominated by (f_k, h_k) . Figure 15.6 shows a filter where each pair (f_l, h_l) in the filter is represented as a black dot. Every point in the filter creates an (infinite) rectangular region, and their union defines the set of pairs not acceptable to the filter. More specifically, a trial point x^+ is acceptable to the filter if (f^+, h^+) lies below or to the left of the solid line in Figure 15.6.

To compare the filter and merit function approaches, we plot in Figure 15.7 the contour line of the set of pairs (f, h) such that $f + \mu h = f_k + \mu h_k$, where x_k is the current iterate. The region to the left of this line corresponds to the set of pairs that reduce the merit function $\phi(x; \mu) = f(x) + \mu h(x)$; clearly this set is quite different from the set of points acceptable to the filter.

If a trial step $x^+ = x_k + \alpha_k p_k$ generated by a line search method gives a pair (f^+, h^+) that is acceptable to the filter, we set $x_{k+1} = x^+$; otherwise, a backtracking line search is performed. In a trust-region method, if the step is not acceptable to the filter, the trust region is reduced, and a new step is computed.

Several enhancements to this filter technique are needed to obtain global convergence and good practical performance. We need to ensure, first of all, that we do not accept a point whose (f, h) pair is very close to the current pair (f_k, h_k) or to another pair in the filter. We

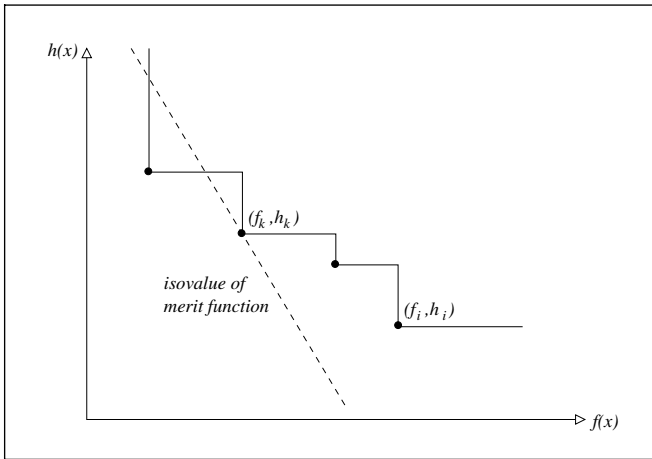


Figure 15.7 Comparing the filter and merit function techniques.

do so by modifying the acceptability criterion and imposing a sufficient decrease condition. A trial iterate x^+ is acceptable to the filter if, for all pairs (f_j, h_j) in the filter, we have that

$$f(x^+) \leq f_j - \beta h_j \quad \text{or} \quad h(x^+) \leq h_j - \beta h_j, \quad (15.33)$$

for $\beta \in (0, 1)$. Although this condition is effective in practice using, say $\beta = 10^{-5}$, for purposes of analysis it may be advantageous to replace the first inequality by

$$f(x^+) \leq f_j - \beta h^+.$$

A second enhancement addresses some problematic aspects of the filter mechanism. Under certain circumstances, the search directions generated by line search methods may require arbitrarily small steplengths α_k to be acceptable to the filter. This phenomenon can cause the algorithm to stall and fail. To guard against this situation, if the backtracking line search generates a steplength that is smaller than a given threshold α_{\min} , the algorithm switches to a *feasibility restoration phase*, which we describe below. Similarly, in a trust-region method, if a sequence of trial steps is rejected by the filter, the trust-region radius may be decreased so much that the trust-region subproblem becomes infeasible (see Section 18.5). In this case, too, the feasibility restoration phase is invoked. (Other mechanisms could be employed to handle this situation, but as we discuss below, the feasibility restoration phase can help the algorithm achieve other useful goals.)

The feasibility restoration phase aims exclusively to reduce the constraint violation, that is, to find an approximate solution to the problem

$$\min_x h(x).$$

Although $h(x)$ defined by (15.31) is not smooth, we show in Chapter 17 how to minimize it using a smooth constrained optimization subproblem. This phase terminates at an iterate that has a sufficiently small value of h and is compatible with the filter.

We now present a framework for filter methods that assumes that iterates are generated by a trust-region method; see Section 18.5 for a discussion of trust-region methods for constrained optimization.

Algorithm 15.1 (General Filter Method).

```

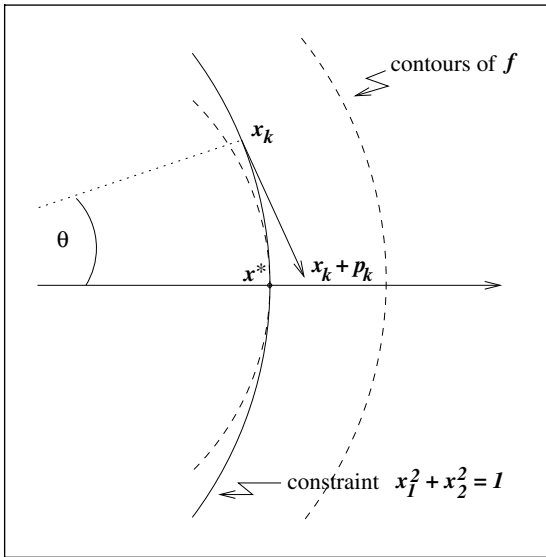
Choose a starting point  $x_0$  and an initial trust-region radius  $\Delta_0$ ;
Set  $k \leftarrow 0$ ;
repeat until a convergence test is satisfied
  if the step-generation subproblem is infeasible
    Compute  $x_{k+1}$  using the feasibility restoration phase;
  else
    Compute a trial iterate  $x^+ = x_k + p_k$ ;
    if  $(f^+, h^+)$  is acceptable to the filter
      Set  $x_{k+1} = x^+$  and add  $(f_{k+1}, h_{k+1})$  to the filter;
      Choose  $\Delta_{k+1}$  such that  $\Delta_{k+1} \geq \Delta_k$ ;
      Remove all pairs from the filter that are dominated
        by  $(f_{k+1}, h_{k+1})$ ;
    else
      Reject the step, set  $x_{k+1} = x_k$ ;
      Choose  $\Delta_{k+1} < \Delta_k$ ;
    end if
  end if
   $k \leftarrow k + 1$ ;
end repeat

```

Other enhancements of this simple filter framework are used in practice; they depend on the choice of algorithm and will be discussed in subsequent chapters.

15.5 THE MARATOS EFFECT

Some algorithms based on merit functions or filters may fail to converge rapidly because they reject steps that make good progress toward a solution. This undesirable phenomenon is often called the *Maratos effect*, because it was first observed by Maratos [199]. It is illustrated by the following example, in which steps p_k , which would yield quadratic convergence if accepted, cause an increase both in the objective function value and the constraint violation.

**Figure 15.8**

Maratos Effect: Example 15.4. Note that the constraint is no longer satisfied after the step from x_k to $x_k + p_k$, and the objective value has increased.

□ **EXAMPLE 15.4** (POWELL [255])

Consider the problem

$$\min f(x_1, x_2) = 2(x_1^2 + x_2^2 - 1) - x_1, \quad \text{subject to} \quad x_1^2 + x_2^2 - 1 = 0. \quad (15.34)$$

One can verify (see Figure 15.8) that the optimal solution is $x^* = (1, 0)^T$, that the corresponding Lagrange multiplier is $\lambda^* = \frac{3}{2}$, and that $\nabla_{xx}^2 \mathcal{L}(x^*, \lambda^*) = I$.

Let us consider an iterate x_k of the form $x_k = (\cos \theta, \sin \theta)^T$, which is feasible for any value of θ . Suppose that our algorithm computes the following step:

$$p_k = \begin{pmatrix} \sin^2 \theta \\ -\sin \theta \cos \theta \end{pmatrix}, \quad (15.35)$$

which yields a trial point

$$x_k + p_k = \begin{pmatrix} \cos \theta + \sin^2 \theta \\ \sin \theta (1 - \cos \theta) \end{pmatrix}.$$

By using elementary trigonometric identities, we have that

$$\|x_k + p_k - x^*\|_2 = 2 \sin^2(\theta/2), \quad \|x_k - x^*\|_2 = 2 |\sin(\theta/2)|,$$

and therefore

$$\frac{\|x_k + p_k - x^*\|_2}{\|x_k - x^*\|_2^2} = \frac{1}{2}.$$

Hence, this step approaches the solution at a rate consistent with Q-quadratic convergence. However, we have that

$$\begin{aligned} f(x_k + p_k) &= \sin^2 \theta - \cos \theta > -\cos \theta = f(x_k), \\ c(x_k + p_k) &= \sin^2 \theta > c(x_k) = 0, \end{aligned}$$

so that, as can be seen in Figure 15.8, both the objective function value and the constraint violation increase over this step. This behavior occurs for any nonzero value of θ , even if the initial point is arbitrarily close to the solution. □

On the example above, any algorithm that requires reduction of a merit function of the form

$$\phi(x; \mu) = f(x) + \mu h(c(x)),$$

where $h(\cdot)$ is a nonnegative function satisfying $h(0) = 0$, will reject the good step (15.35). (Examples of such merit functions include the ϕ_1 and ϕ_2 penalty functions.) The step (15.35) will also be rejected by the filter mechanism described above because the pair $(f(x_k + p_k), h(x_k + p_k))$ is dominated by (f_k, h_k) . Therefore, all these approaches will suffer from the Maratos effect.

If no remedial measures are taken, the Maratos effect can slow optimization methods by interfering with good steps away from the solution and by preventing superlinear convergence. Strategies for avoiding the Maratos effect include the following.

1. We can use a merit function that does not suffer from the Maratos effect. An example is Fletcher's augmented Lagrangian function (15.26).
2. We can use a second-order correction in which we add to p_k a step \hat{p}_k , which is computed at $c(x_k + p_k)$ and which decreases the constraint violation.
3. We can allow the merit function ϕ to increase on certain iterations; that is, we can use a nonmonotone strategy.

We discuss the last two approaches in the next section.

15.6 SECOND-ORDER CORRECTION AND NONMONOTONE TECHNIQUES

By adding a correction term that decreases the constraint violation, various algorithms are able to overcome the difficulties associated with the Maratos effect. We describe this technique with respect to the equality-constrained problem, in which the constraint is $c(x) = 0$, where $c : \mathbb{R}^n \rightarrow \mathbb{R}^{|\mathcal{E}|}$.

Given a step p_k , the second-order correction step \hat{p}_k is defined to be

$$\hat{p}_k = -A_k^T (A_k A_k^T)^{-1} c(x_k + p_k), \quad (15.36)$$

where $A_k = A(x_k)$ is the Jacobian of c at x_k . Note that \hat{p}_k has the property that it satisfies a linearization of the constraint c at the point $x_k + p_k$, that is,

$$A_k \hat{p}_k + c(x_k + p_k) = 0.$$

In fact, \hat{p}_k is the minimum-norm solution of this equation. (A different interpretation of the second-order correction is given in Section 18.3.)

The effect of the correction step \hat{p}_k is to decrease the quantity $\|c(x)\|$ to the order of $\|x_k - x^*\|^3$, provided the primary step p_k satisfies $A_k p_k + c(x_k) = 0$. This estimate indicates that the step from x_k to $x_k + p_k + \hat{p}_k$ will decrease the merit function, at least near the solution. The cost of this enhancement includes the additional evaluation of the constraint function c at $x_k + p_k$ and the linear algebra required to calculate the step \hat{p}_k from (15.36).

We now describe an algorithm that uses a merit function together with a line-search strategy and a second-order correction step. We assume that the search direction p_k and the penalty parameter μ_k are computed so that p_k is a descent direction for the merit function, that is, $D(\phi(x_k; \mu); p_k) < 0$. In Chapters 18 and 19, we discuss how to accomplish these goals. The key feature of the algorithm is that, if the full step $\alpha_k = 1$ does not produce satisfactory descent in the merit function, we try the second-order correction step *before* backtracking along the original direction p_k .

Algorithm 15.2 (Generic Algorithm with Second-Order Correction).

Choose parameters $\eta \in (0, 0.5)$ and τ_1, τ_2 with $0 < \tau_1 < \tau_2 < 1$;

Choose initial point x_0 ; set $k \leftarrow 0$;

repeat until a convergence test is satisfied:

 Compute a search direction p_k ;

 Set $\alpha_k \leftarrow 1$, **newpoint** \leftarrow **false**;

while **newpoint** = **false**

if $\phi(x_k + \alpha_k p_k; \mu) \leq \phi(x_k; \mu) + \eta \alpha_k D(\phi(x_k; \mu); p_k)$

 Set $x_{k+1} \leftarrow x_k + \alpha_k p_k$;

 Set **newpoint** \leftarrow **true**;

```

else if  $\alpha_k = 1$ 
    Compute  $\hat{p}_k$  from (15.36);
    if  $\phi(x_k + p_k + \hat{p}_k; \mu) \leq \phi(x_k; \mu) + \eta D(\phi(x_k; \mu); p_k)$ 
        Set  $x_{k+1} \leftarrow x_k + p_k + \hat{p}_k$ ;
        Set newpoint  $\leftarrow$  true;
    else
        Choose new  $\alpha_k$  in  $[\tau_1\alpha_k, \tau_2\alpha_k]$ ;
    end
else
    Choose new  $\alpha_k$  in  $[\tau_1\alpha_k, \tau_2\alpha_k]$ ;
end
end while
end repeat

```

In this algorithm, the full second-order correction step \hat{p}_k is discarded if it does not produce a reduction in the merit function. We do not backtrack along the direction $p_k + \hat{p}_k$ because it is not guaranteed to be a descent direction for the merit function. A variation of this algorithm applies the second-order correction step only if the sufficient decrease condition (15.29) is violated as a result of an increase in the norm of the constraints.

The second-order correction strategy is effective in practice. The cost of performing the extra constraint function evaluation and an additional backsolve in (15.36) is outweighed by added robustness and efficiency.

NONMONOTONE (WATCHDOG) STRATEGY

The inefficiencies caused by the Maratos effect can also be avoided by occasionally accepting steps that increase the merit function; such steps are called *relaxed steps*. There is a limit to our tolerance, however. If a sufficient reduction of the merit function has not been obtained within a certain number of iterates of the relaxed step ($\hat{\ell}$ iterates, say), then we return to the iterate before the relaxed step and perform a normal iteration, using a line search or some other technique to force a reduction in the merit function.

In contrast with the second-order correction, which aims only to improve satisfaction of the constraints, this nonmonotone strategy always takes regular steps p_k of the algorithm that aim both for improved feasibility and optimality. The hope is that any increase in the merit function over a single step will be temporary, and that subsequent steps will more than compensate for it.

We now describe a particular instance of the nonmonotone approach called the *watchdog strategy*. We set $\hat{\ell} = 1$, so that we allow the merit function to increase on just a single step before insisting on a sufficient decrease in the merit function. As above, we focus our discussion on a line search algorithm that uses a nonsmooth merit function ϕ . We assume that the penalty parameter μ is not changed until a successful cycle has been

completed. To simplify the notation, we omit the dependence of ϕ on μ and write the merit function as $\phi(x)$ and the directional derivative as $D(\phi(x); p_k)$.

Algorithm 15.3 (Watchdog).

Choose a constant $\eta \in (0, 0.5)$ and an initial point x_0 ;

Set $k \leftarrow 0, \mathcal{S} \leftarrow \{0\}$;

repeat until a termination test is satisfied

 Compute a step p_k ;

 Set $x_{k+1} \leftarrow x_k + p_k$;

if $\phi(x_{k+1}) \leq \phi(x_k) + \eta D(\phi(x_k); p_k)$

$k \leftarrow k + 1, \mathcal{S} \leftarrow \mathcal{S} \cup \{k\}$;

else

 Compute a search direction p_{k+1} from x_{k+1} ;

 Find α_{k+1} such that

$$\phi(x_{k+2}) \leq \phi(x_{k+1}) + \eta \alpha_{k+1} D(\phi(x_{k+1}); p_{k+1});$$

 Set $x_{k+2} \leftarrow x_{k+1} + \alpha_{k+1} p_{k+1}$;

if $\phi(x_{k+1}) \leq \phi(x_k)$ **or** $\phi(x_{k+2}) \leq \phi(x_k) + \eta D(\phi(x_k); p_k)$

$k \leftarrow k + 2, \mathcal{S} \leftarrow \mathcal{S} \cup \{k\}$;

else if $\phi(x_{k+2}) > \phi(x_k)$

 (* return to x_k and search along p_k *)

 Find α_k such that $\phi(x_{k+3}) \leq \phi(x_k) + \eta \alpha_k D(\phi(x_k); p_k)$;

 Compute $x_{k+3} = x_k + \alpha_k p_k$;

$k \leftarrow k + 3, \mathcal{S} \leftarrow \mathcal{S} \cup \{k\}$;

else

 Compute a direction p_{k+2} from x_{k+2} ;

 Find α_{k+2} such that

$$\phi(x_{k+3}) \leq \phi(x_{k+2}) + \eta \alpha_{k+2} D(\phi(x_{k+2}); p_{k+2});$$

 Set $x_{k+3} \leftarrow x_{k+2} + \alpha_{k+2} p_{k+2}$;

$k \leftarrow k + 3, \mathcal{S} \leftarrow \mathcal{S} \cup \{k\}$;

end

end

end (repeat)

The set \mathcal{S} is not required by the algorithm and is introduced only to identify the iterates for which a sufficient merit function reduction was obtained. Note that at least a third of the iterates have their indices in \mathcal{S} . By using this fact, one can show that various constrained optimization methods that use the watchdog technique are globally convergent. One can also show that for all sufficiently large k , the step length is $\alpha_k = 1$ and the convergence rate is superlinear.


In practice, it may be advantageous to allow increases in the merit function for more than one iteration. Values of \hat{t} such as 5 or 8 are typical. As this discussion indicates, careful implementations of the watchdog technique have a certain degree of complexity, but


the added complexity is worthwhile because the approach has good practical performance. A potential advantage of the watchdog technique over the second-order correction strategy is that it may require fewer evaluations of the constraint functions. In the best case, most of the steps will be full steps, and there will rarely be a need to return to an earlier point.


NOTES AND REFERENCES

Techniques for eliminating linear constraints are described, for example, in Fletcher [101] and Gill, Murray, and Wright [131]. For a thorough discussion of merit functions see Boggs and Tolle [33] and Conn, Gould, and Toint [74]. Some of the earliest references on nonmonotone methods include Grippo, Lampariello and Lucidi [158], and Chamberlain et al [57]; see [74] for a review of nonmonotone techniques and an extensive list of references. The concept of a filter was introduced by Fletcher and Leyffer [105]; our discussion of filters is based on that paper. Second-order correction steps are motivated and discussed in Fletcher [101].

EXERCISES

 **15.1** In Example 15.1, consider these three choices of the working set: $\mathcal{W} = \{3\}$, $\mathcal{W} = \{1, 2\}$, $\mathcal{W} = \{2, 3\}$. Show that none of these working sets are the optimal active set for (15.5).


 **15.2** For the problem in Example 15.3, perform simple elimination of the variables x_2 and x_5 to obtain an unconstrained problem in the remaining variables x_1, x_3, x_4 , and x_6 . Similarly to (15.12), express the eliminated variables explicitly in terms of the retained variables.


 **15.3** Do the following problems have solutions? Explain.


$$\min x_1 + x_2 \quad \text{subject to } x_1^2 + x_2^2 = 2, \quad 0 \leq x_1 \leq 1, \quad 0 \leq x_2 \leq 1;$$


$$\min x_1 + x_2 \quad \text{subject to } x_1^2 + x_2^2 \leq 1, \quad x_1 + x_2 = 3;$$

$$\min x_1 x_2 \quad \text{subject to } x_1 + x_2 = 2.$$

 **15.4** Show that if in Example 15.2 we eliminate x in terms of y , then the correct solution of the problem is obtained by performing unconstrained minimization.

 **15.5** Show that the basis matrices (15.15) are linearly independent.

 **15.6** Show that the particular solution $Q_1 R^{-T} \Pi^T b$ of $Ax = b$ is identical to $A^T (AA^T)^{-1} b$.


 **15.7** In this exercise we compute basis matrices that attempt to compromise between the orthonormal basis (15.22) and simple elimination (15.15). We assume that the basis matrix is given by the first m columns of A , so that $P = I$ in (15.7), and define

$$Y = \begin{bmatrix} I \\ (B^{-1}N)^T \end{bmatrix}, \quad Z = \begin{bmatrix} -B^{-1}N \\ I \end{bmatrix}. \quad (15.37)$$

- (a) Show that the columns of Y and Z are no longer of norm 1 and that the relations $AZ = 0$ and $Y^T Z = 0$ hold. Therefore, the columns of Y and Z form a linearly independent set, showing that (15.37) is a valid choice of the basis matrices.
- (b) Show that the particular solution $Y(AY)^{-1}b$ defined by this choice of Y is, as in the orthogonal factorization approach, the minimum-norm solution of $Ax = b$. More specifically, show that

$$Y(AY)^{-1} = A^T(AA^T)^{-1}.$$

It follows that the matrix $Y(AY)^{-1}$ is independent of the choice of basis matrix B in (15.7), and its conditioning is determined by that of A alone. (Note, however, that the matrix Z still depends explicitly on B , so a careful choice of B is needed to ensure well conditioning in this part of the computation.)

 **15.8** Verify that by adding the inequality constraint $3x_1 + 2x_3 \geq 1$ to the problem (15.11), the elimination (15.12) transforms the problem into one of minimizing the function (15.13) subject to the inequality constraint (15.23).