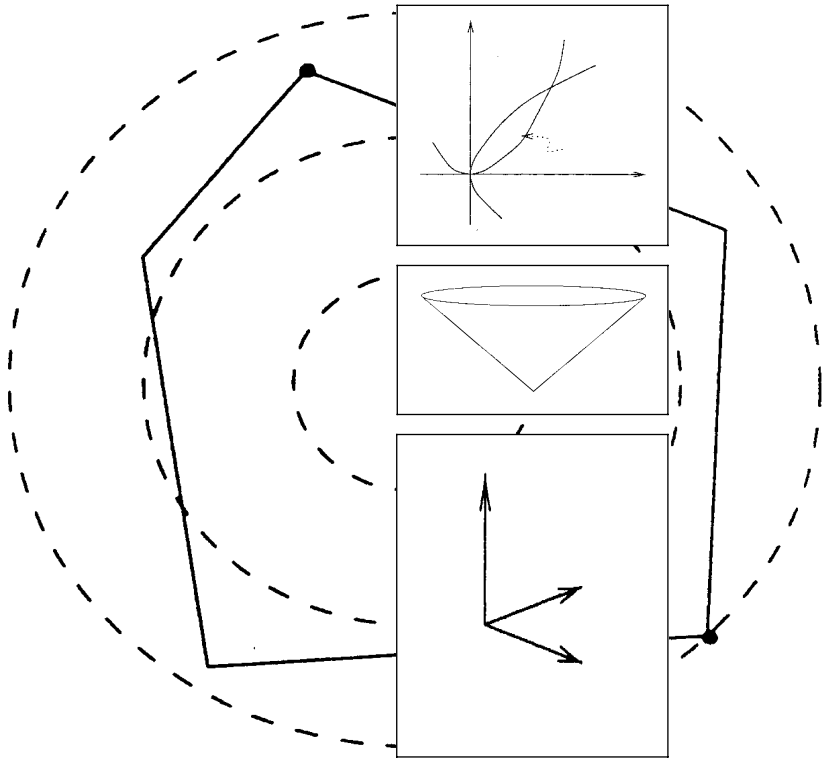
APPENDIX A



Background Material

A.1 ELEMENTS OF ANALYSIS, GEOMETRY, TOPOLOGY

TOPOLOGY OF THE EUCLIDEAN SPACE $\mathbb{R}^n$

Let $\mathcal{F}$ be a subset of $\mathbb{R}^n$, and suppose that $\{x_k\}$ is a sequence of points belonging to $\mathcal{F}$. We say that a sequence $\{x_k\}$ *converges* to some point x, written $\lim_{k\to\infty} x_k = x$, if for any $\epsilon > 0$, there is an index K such that

$$\|x_k - x\| \leq \epsilon, \qquad \text{for all } k \geq K.$$

For example, the sequence $\{x_k\}$ defined by $x_k = (1-2^{-k}, 1/k^2)^T$ converges to $(1, 0)^T$. Other examples of convergent sequences are given in Chapter 2, where the *rate of convergence* is also discussed.

We say that $\hat{x} \in \mathbb{R}^n$ is an *accumulation point* or *limit point* for $\{x_k\}$ if there is some infinite subsequence of indices $k_1, k_2, k_3, \ldots$ such that

$$\lim_{i\to\infty} x_{k_i} = \hat{x}.$$

Another way to define a limit point is to say that for any $\epsilon > 0$ and all positive integers K, we have

$$\|x_k - x\| \leq \epsilon, \quad \text{for some } k \geq K.$$

An example is given by the sequence

$$\begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1/2 \\ 1/2 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1/4 \\ 1/4 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1/8 \\ 1/8 \end{bmatrix}, \ldots, \tag{A.1}$$

which has exactly two limit points: $\hat{x} = (0, 0)^T$ and $\hat{x} = (1, 1)^T$. A sequence can even have an infinite number of limit points. An example is the sequence $x_k = \sin k$, for which every point in the interval $[-1, 1]$ is a limit point.

If $\{t_k\}$ is a sequence of *real* numbers, we can define two other concepts: the lim inf and lim sup. The lim inf is the smallest accumulation point, while the lim sup is the largest. That is, we write $\hat{t} = \limsup_{k\to\infty} t_k$ if the following two conditions hold:

(i) there is an infinite subsequence $k_1, k_2, \ldots$ such that $\lim_{i\to\infty} t_{k_i} = \hat{t}$, and

(ii) there is no other limit point t such that $t > \hat{t}$.

The sequence $1, \frac{1}{2}, 1, \frac{1}{4}, 1, \frac{1}{8}, \ldots$ has a lim inf of 0 and a lim sup of 1.

The set $\mathcal{F}$ is *bounded* if there is some real number $M > 0$ such that

$$\|x\| \leq M, \quad \text{for all } x \in \mathcal{F}.$$

A subset $\mathcal{F} \subset \mathbb{R}^n$ is *open* if for every $x \in \mathcal{F}$, we can find a positive number $\epsilon > 0$ such that the ball of radius ϵ around x is contained in $\mathcal{F}$; that is,

$$\{y \in \mathbb{R}^n \mid \|y - x\| \leq \epsilon\} \subset \mathcal{F}.$$

The set $\mathcal{F}$ is *closed* if for all possible sequences of points $\{x_k\}$ in $\mathcal{F}$, all limit points of $\{x_k\}$ are elements of $\mathcal{F}$. For instance, the set $\mathcal{F} = (0, 1) \cup (2, 10)$ is an open subset of $\mathbb{R}$, while $\mathcal{F} = [0, 1] \cup [2, 5]$ is a closed subset of $\mathbb{R}$. The set $\mathcal{F} = (0, 1]$ is a subset of $\mathbb{R}$ that is neither open nor closed.

The *interior* of a set $\mathcal{F}$, denoted by $\text{int}\mathcal{F}$, is the largest open set contained in $\mathcal{F}$. The *closure* of $\mathcal{F}$, denoted by $\text{cl}\mathcal{F}$, is the smallest closed set containing $\mathcal{F}$. In other words, we have

$$x \in \text{cl}\mathcal{F} \quad \text{if } \lim_{k\to\infty} x_k = x \text{ for some sequence } \{x_k\} \text{ of points in } \mathcal{F}.$$

If $\mathcal{F} = (-1, 1] \cup [2, 4)$, then

$$\text{cl}\mathcal{F} = [-1, 1] \cup [2, 4], \quad \text{int}\mathcal{F} = (-1, 1) \cup (2, 4).$$

Note that if $\mathcal{F}$ is open, then $\text{int}\mathcal{F} = \mathcal{F}$, while if $\mathcal{F}$ is closed, then $\text{cl}\mathcal{F} = \mathcal{F}$.

The set $\mathcal{F}$ is *compact* if every sequence $\{x^k\}$ of points in $\mathcal{F}$ has at least one limit point, and all such limit points are in $\mathcal{F}$. (This definition is equivalent to the more formal one involving covers of $\mathcal{F}$.) The following is a central result in topology:

$\mathcal{F} \in \mathbb{R}^n$ is closed and bounded $\Rightarrow$ $\mathcal{F}$ is compact.

Given a point $x \in \mathbb{R}^n$, we call $\mathcal{N} \in \mathbb{R}^n$ a *neighborhood of* x if it is an open set containing x. An especially useful neighborhood is the *open ball of radius* ϵ *around* x, which is denoted by $\mathbb{B}(x, \epsilon)$; that is,

$$\mathbb{B}(x, \epsilon) = \{y \mid \|y - x\| < \epsilon\}.$$

A *cone* is a set $\mathcal{F}$ with the property that for all $x \in \mathcal{F}$ we have

$$x \in \mathcal{F} \;\Rightarrow\; \alpha x \in \mathcal{F}, \quad \text{for all } \alpha \geq 0. \tag{A.2}$$

For instance, the set $\mathcal{F} \subset \mathbb{R}^2$ defined by

$$\{(x_1, x_2) \mid x_1 > 0,\ x_2 \geq 0\}$$

is a cone in $\mathbb{R}^2$.

Finally, we define the affine hull and relative interior of a set. Given $\mathcal{F} \subset \mathbb{R}^n$, the affine hull is the smallest affine set containing $\mathcal{F}$, that is,

$$\text{aff}\mathcal{F} = \{x \mid x \text{ is a linear combination of vectors in } \mathcal{F}\}. \tag{A.3}$$

For instance, when $\mathcal{F}$ is the "ice-cream cone" defined as below by (A.26), we have $\text{aff}\mathcal{F} = \mathbb{R}^3$, while if $\mathcal{F}$ is the set of two isolated points $\mathcal{F} = \{(1, 0, 0), (0, 2, 0)\}$, we have

$$\text{aff}\mathcal{F} = \{(x_1, x_2, 0) \mid \text{for all } x_1 \text{ and } x_2\}.$$

The *relative interior* $\text{ri}\mathcal{F}$ of the set S is its *interior relative to* $\text{aff}\mathcal{F}$. If $x \in \mathcal{F}$, then $x \in \text{ri}\mathcal{F}$ if there is an $\epsilon > 0$ such that

$$(x + \epsilon\mathcal{B}) \cap \text{aff}\mathcal{F} \subset \mathcal{F}.$$

Referring again to the ice-cream cone (A.26), we have that

$$\text{ri}\mathcal{F} = \left\{ x \in \mathbb{R}^3 \;\middle|\; x_3 > 2\sqrt{x_1^2 + x_2^2} \right\}.$$

For the set of two isolated points $\mathcal{F} = \{(1, 0, 0), (0, 2, 0)\}$, we have $\mathrm{ri}\mathcal{F} = \emptyset$. For the set $\mathcal{F}$ defined by

$$\mathcal{F} \stackrel{\text{def}}{=} \{x \in \mathbb{R}^3 \,|\, x_1 \in [0, 1],\ x_2 \in [0, 1],\ x_3 = 0\},$$

we have that

$$\mathrm{aff}\mathcal{F} = \mathbb{R} \times \mathbb{R} \times \{0\}, \qquad \mathrm{ri}\mathcal{F} = \{x \in \mathbb{R}^3 \,|\, x_1 \in (0, 1),\ x_2 \in (0, 1),\ x_3 = 0\}.$$

CONTINUITY AND LIMITS

Let f be a function that maps some domain $\mathcal{D} \subset \mathbb{R}^n$ to the space $\mathbb{R}^m$. For some point $x_0 \in \mathrm{cl}\mathcal{D}$, we write

$$\lim_{x \to x_0} f(x) = f_0 \tag{A.4}$$

(spoken "the limit of $f(x)$ as x approaches x_0 is f_0") if for all $\epsilon > 0$, there is a value $\delta > 0$ such that

$$\|x - x_0\| < \delta \ \text{ and } \ x \in \mathcal{D} \ \Rightarrow \ \|f(x) - f_0\| < \epsilon.$$

We say that f is *continuous* at x_0 if $x_0 \in \mathcal{D}$ and the expression (A.4) holds with $f_0 = f(x_0)$. We say that f is continuous on its domain $\mathcal{D}$ if f is continuous for all $x_0 \in \mathcal{D}$.

An example is provided by the function

$$f(x) = \begin{cases} -x & \text{if } x \in [-1, 1], x \neq 0, \\ 5 & \text{for all other } x \in [-10, 10]. \end{cases} \tag{A.5}$$

This function is defined on the domain $[-10, 10]$ and is continuous at all points of the domain except the points $x = 0$, $x = 1$, and $x = -1$. At $x = 0$, the expression (A.4) holds with $f_0 = 0$, but the function is not continuous at this point because $f_0 \neq f(0) = 5$. At $x = -1$, the limit (A.4) is not defined, because the function values in the neighborhood of this point are close to both 5 and -1, depending on whether x is slightly smaller or slightly larger than -1. Hence, the function is certainly not continuous at this point. The same comments apply to the point $x = 1$.

In the special case of $n = 1$ (that is, the argument of f is a real scalar), we can also define the *one-sided limit*. Given $x_0 \in \mathrm{cl}\mathcal{D}$, We write

$$\lim_{x \downarrow x_0} f(x) = f_0 \tag{A.6}$$

(spoken "the limit of $f(x)$ as x approaches x_0 from above is f_0") if for all $\epsilon > 0$, there is a value $\delta > 0$ such that

$$x_0 < x < x_0 + \delta \quad \text{and} \quad x \in \mathcal{D} \quad \Rightarrow \quad \|f(x) - f_0\| < \epsilon.$$

Similarly, we write

$$\lim_{x \uparrow x_0} f(x) = f_0 \tag{A.7}$$

(spoken "the limit of $f(x)$ as x approaches x_0 from below is f_0") if for all $\epsilon > 0$, there is a value $\delta > 0$ such that

$$x_0 - \delta < x < x_0 \quad \text{and} \quad x \in \mathcal{D} \quad \Rightarrow \quad \|f(x) - f_0\| < \epsilon.$$

For the function defined in (A.5), we have that

$$\lim_{x \downarrow 1} f(x) = 5, \qquad \lim_{x \uparrow 1} f(x) = 1.$$

The function f is said to be *Lipschitz continuous* if there is a constant $M > 0$ such that for any two points x_0, x_1 in $\mathcal{D}$, we have

$$\|f(x_1) - f(x_0)\| \leq M\|x_1 - x_0\|. \tag{A.8}$$

The function f is *locally Lipschitz continuous* at a point $x_0 \in \text{int}\mathcal{D}$ if there is some neighborhood $\mathcal{N}$ with $x_0 \in \mathcal{N} \subset \mathcal{D}$ such that the property (A.8) holds for all x_0 and x_1 in $\mathcal{N}$.

DERIVATIVES

Let $\phi : \mathbb{R} \to \mathbb{R}$ be a real-valued function of a real variable (sometimes known as a *univariate* function). The first derivative $\phi'(\alpha)$ is defined by

$$\frac{d\phi}{d\alpha} = \phi'(\alpha) \stackrel{\text{def}}{=} \lim_{\epsilon \to 0} \frac{\phi(\alpha + \epsilon) - \phi(\alpha)}{\epsilon}. \tag{A.9}$$

The second derivative is obtained by substituting ϕ by ϕ' in this same formula; that is,

$$\frac{d^2\phi}{d\alpha^2} = \phi''(\alpha) \stackrel{\text{def}}{=} \lim_{\epsilon \to 0} \frac{\phi'(\alpha + \epsilon) - \phi'(\alpha)}{\epsilon}. \tag{A.10}$$

(We assume implicitly that ϕ is smooth enough that these limits exist.) Suppose now that α in turn depends on another quantity β (we denote this dependence by writing $\alpha = \alpha(\beta)$).

We can use the *chain rule* to calculate the derivative of ϕ with respect to β:

$$\frac{d\phi(\alpha(\beta))}{d\beta} = \frac{d\phi}{d\alpha}\frac{d\alpha}{d\beta} = \phi'(\alpha)\alpha'(\beta). \tag{A.11}$$

Consider now the function $f : \mathbb{R}^n \to \mathbb{R}$, which is a real-valued function of n independent variables. We typically gather the variables into a vector x, which we can write componentwise as $x = (x_1, x_2, \ldots, x_n)$. The n-vector of first derivatives of f—the *gradient*—is defined as

$$\nabla f(x) = \begin{bmatrix} \dfrac{\partial f}{\partial x_1} \\ \vdots \\ \dfrac{\partial f}{\partial x_n} \end{bmatrix}. \tag{A.12}$$

(The notation "∇" is used frequently in the optimization literature to denote the first derivative. In cases of ambiguity, a subscript such as "x" or "t" is added to ∇ to indicate the variable with respect to which the differentiation takes place.) Each partial derivative $\partial f/\partial x_i$ measures the sensitivity of the function to just one of the components of x; that is,

$$\begin{aligned} &\partial f/\partial x_i \\ &\stackrel{\text{def}}{=} \lim_{\epsilon \to 0} \frac{f(x_1, \ldots, x_{i-1}, x_i + \epsilon, x_{i+1}, \ldots, x_n) - f(x_1, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots, x_n)}{\epsilon} \\ &= \frac{f(x + \epsilon e_i) - f(x)}{\epsilon}, \end{aligned}$$

where e_i is the vector $(0, 0, \ldots, 0, 1, 0, \ldots, 0)$, where the 1 appears in the ith position. The matrix of second partial derivatives of f is known as the *Hessian*, and is defined as

$$\nabla^2 f(x) = \begin{bmatrix} \dfrac{\partial^2 f}{\partial x_1^2} & \dfrac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \dfrac{\partial^2 f}{\partial x_1 \partial x_n} \\ \dfrac{\partial^2 f}{\partial x_2 \partial x_1} & \dfrac{\partial^2 f}{\partial x_2^2} & \cdots & \dfrac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & & \vdots \\ \dfrac{\partial^2 f}{\partial x_n \partial x_1} & \dfrac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \dfrac{\partial^2 f}{\partial x_n^2} \end{bmatrix}.$$

We say that f is *differentiable* if all first partial derivatives of f exist, and *continuously differentiable* if in addition these derivatives are continuous functions of x. Similarly, f is *twice differentiable* if all second partial derivatives of f exist and *twice continuously differentiable* if

they are also continuous. Note that when f is twice continuously differentiable, the Hessian is a symmetric matrix, since

$$\frac{\partial^2 f}{\partial x_i \partial x_j} = \frac{\partial^2 f}{\partial x_j \partial x_i}, \quad \text{for all } i, j = 1, 2, \ldots, n.$$

When the vector x in turn depends on another vector t (that is, $x = x(t)$), the chain rule (A.11) for the univariate function can be extended as follows:

$$\nabla_t f(x(t)) = \sum_{i=1}^{n} \frac{\partial f}{\partial x_i} \nabla x_i(t). \tag{A.13}$$

DIRECTIONAL DERIVATIVES

If f is continuously differentiable and $p \in \mathbb{R}^n$, then the *directional derivative* of f in the direction p is given by

$$D(f(x); p) \stackrel{\text{def}}{=} \lim_{\epsilon \to 0} \frac{f(x + \epsilon p) - f(x)}{\epsilon} = \nabla f(x)^T p. \tag{A.14}$$

To verify this formula, we define the function

$$\phi(\alpha) = f(x + \alpha p) = f(y(\alpha)), \tag{A.15}$$

where $y(\alpha) = x + \alpha p$. Note that

$$\lim_{\epsilon \to 0} \frac{f(x + \epsilon p) - f(x)}{\epsilon} = \lim_{\epsilon \to 0} \frac{\phi(\epsilon) - \phi(0)}{\epsilon} = \phi'(0).$$

By applying the chain rule (A.13) to $f(y(\alpha))$, we obtain

$$\begin{aligned} \phi'(\alpha) &= \sum_{i=1}^{n} \frac{\partial f(y(\alpha))}{\partial y_i} \nabla y_i(\alpha) \\ &= \sum_{i=1}^{n} \frac{\partial f(y(\alpha))}{\partial y_i} p_i = \nabla f(y(\alpha))^T p = \nabla f(x + \alpha p)^T p. \end{aligned} \tag{A.16}$$

We obtain (A.14) by setting $\alpha = 0$ and comparing the last two expressions.

The directional derivative is sometimes defined even when the function f itself is not differentiable. For instance, if $f(x) = \|x\|_1$, we have from the definition (A.14) that

$$D(\|x\|_1; p) = \lim_{\epsilon \to 0} \frac{\|x + \epsilon p\|_1 - \|x\|_1}{\epsilon} = \lim_{\epsilon \to 0} \frac{\sum_{i=1}^{n} |x_i + \epsilon p_i| - \sum_{i=1}^{n} |x_i|}{\epsilon}.$$

If $x_i > 0$, we have $|x_i + \epsilon p_i| = |x_i| + \epsilon p_i$ for all ϵ sufficiently small. If $x_i < 0$, we have $|x_i + \epsilon p_i| = |x_i| - \epsilon p_i$, while if $x_i = 0$, we have $|x_i + \epsilon p_i| = \epsilon |p_i|$. Therefore, we have

$$D(\|x\|_1; p) = \sum_{i|x_i<0} -p_i + \sum_{i|x_i>0} p_i + \sum_{i|x_i=0} |p_i|,$$

so the directional derivative of this function exists for any x and p. Its first derivative does *not* exist, however, whenever any of the components of x are zero.

❑ EXAMPLE A.1

Let $f : \mathbb{R}^2 \to \mathbb{R}$ be defined by $f(x_1, x_2) = x_1^2 + x_1 x_2$, where $x_1 = \sin t_1 + t_2^2$ and $x_2 = (t_1 + t_2)^2$. The chain rule (A.13) yields

$$\begin{aligned}
&\nabla_t f(x(t)) \\
&\quad = \sum_{i=1}^{n} \frac{\partial f}{\partial x_i} \nabla x_i(t) \\
&\quad = (2x_1 + x_2) \begin{bmatrix} \cos t_1 \\ 2t_2 \end{bmatrix} + x_1 \begin{bmatrix} 2(t_1 + t_2) \\ 2(t_1 + t_2) \end{bmatrix} \\
&\quad = \left(2\left(\sin t_1 + t_2^2\right) + (t_1 + t_2)^2\right) \begin{bmatrix} \cos t_1 \\ 2t_2 \end{bmatrix} + \left(\sin t_1 + t_2^2\right) \begin{bmatrix} 2(t_1 + t_2) \\ 2(t_1 + t_2) \end{bmatrix}.
\end{aligned}$$

If, on the other hand, we substitute directly for x into the definition of f, we obtain

$$f(x(t)) = \left(\sin t_1 + t_2^2\right)^2 + \left(\sin t_1 + t_2^2\right)(t_1 + t_2)^2.$$

The reader should verify that the gradient of this expression is identical to the one obtained above by applying the chain rule.

MEAN VALUE THEOREM

We now recall the mean value theorem for univariate functions. Given a continuously differentiable function $\phi : \mathbb{R} \to \mathbb{R}$ and two real numbers α_0 and α_1 that satisfy $\alpha_1 > \alpha_0$, we have that

$$\phi(\alpha_1) = \phi(\alpha_0) + \phi'(\xi)(\alpha_1 - \alpha_0) \tag{A.17}$$

for some $\xi \in (\alpha_0, \alpha_1)$. An extension of this result to a multivariate function $f : \mathbb{R}^n \to \mathbb{R}$ is that for any vector p we have

$$f(x + p) = f(x) + \nabla f(x + \alpha p)^T p, \tag{A.18}$$

for some $\alpha \in (0, 1)$. (This result can be proved by defining $\phi(\alpha) = f(x + \alpha p)$, $\alpha_0 = 0$, and $\alpha_1 = 1$ and applying the chain rule, as above.)

❑ EXAMPLE A.2

Consider $f : \mathbb{R}^2 \to \mathbb{R}$ defined by $f(x) = x_1^3 + 3x_1x_2^2$, and let $x = (0, 0)^T$ and $p = (1, 2)^T$. It is easy to verify that $f(x) = 0$ and $f(x + p) = 13$. Since

$$\nabla f(x + \alpha p) = \begin{bmatrix} 3(x_1 + \alpha p_1)^2 + 3(x_2 + \alpha p_2)^2 \\ 6(x_1 + \alpha p_1)(x_2 + \alpha p_2) \end{bmatrix} = \begin{bmatrix} 15\alpha^2 \\ 12\alpha^2 \end{bmatrix},$$

we have that $\nabla f(x + \alpha p)^T p = 39\alpha^2$. Hence the relation (A.18) holds when we set $\alpha = 1/\sqrt{13}$, which lies in the open interval $(0, 1)$, as claimed. ❑

An alternative expression to (A.18) can be stated for twice differentiable functions: We have

$$f(x + p) = f(x) + \nabla f(x)^T p + \frac{1}{2} p^T \nabla^2 f(x + \alpha p)^T p, \tag{A.19}$$

for some $\alpha \in (0, 1)$. In fact, this expression is one form of Taylor's theorem, Theorem 2.1 in Chapter 2, to which we refer throughout the book.

IMPLICIT FUNCTION THEOREM

The implicit function theorem lies behind a number of important results in local convergence theory of optimization algorithms and in the characterization of optimality (see Chapter 12). We present a brief outline here based on the discussion in Lang [147, p. 131].

Theorem A.1 (Implicit Function Theorem).
Let $h : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ be a function such that

(i) *$h(z^*, 0) = 0$ for some $z^* \in \mathbb{R}^n$,*

(ii) *the function $h(\cdot, \cdot)$ is Lipschitz continuously differentiable in some neighborhood of $(z^*, 0)$, and*

(iii) $\nabla_z h(z,t)$ *is nonsingular at the point* $(z,t) = (z^*, 0)$.

Then the function $z : \mathbb{R}^m \to \mathbb{R}^n$ *defined implicitly by* $h(z(t),t) = 0$ *is well-defined and Lipschitz continuous for* $t \in \mathbb{R}^m$ *in some neighborhood of the origin.*

This theorem is frequently applied to parametrized systems of linear equations, in which z is obtained as the solution of

$$M(t)z = g(t),$$

where $M(\cdot) \in \mathbb{R}^{n\times n}$ has $M(0)$ nonsingular, and $g(\cdot) \in \mathbb{R}^n$. To apply the theorem, we define

$$h(z,t) = M(t)z - g(t).$$

If $M(\cdot)$ and $g(\cdot)$ are Lipschitz continuously differentiable in some neighborhood of 0, the theorem implies that $z(t) = M(t)^{-1}g(t)$ is a Lipschitz continuous function of t for all t in some neighborhood of 0.

GEOMETRY OF FEASIBLE SETS

In describing the theory of constrained optimization, we assume for the most part that the set of feasible points is described by algebraic equations. That is, we aim to minimize a function f over the set Ω defined by

$$\Omega = \{x \in \mathbb{R}^n \mid c_i(x) = 0,\ i \in \mathcal{E};\quad c_i(x) \geq 0,\ i \in \mathcal{I}\}, \tag{A.20}$$

where $\mathcal{E}$ and $\mathcal{I}$ are the index sets of equality and inequality constraints, respectively (see also (12.2)). In some situations, however, it is useful to abandon the algebraic description and merely consider the set Ω on its own merits, as a general subset of Ω. By doing this, we no longer tie Ω down to any particular algebraic description. (Note that for any given set Ω, there may be infinitely many collections of constraint functions c_i, $i \in \mathcal{E} \cup \mathcal{I}$, such that the identification (A.20) holds.)

There are advantages and disadvantages to taking the purely geometric viewpoint. By not tying ourselves to a particular algebraic description, we avoid the theoretical complications caused by redundant, linearly dependent, or insufficiently smooth constraints. We also avoid some practical problems such as poor scaling. On the other hand, most of the development of theory, algorithms, and software for constrained optimization assumes that an algebraic description of Ω is available. If we choose instead to work explicitly with Ω, we must reformulate the optimality conditions and algorithms in terms of this set, without reference to its algebraic description. Many such tools are available, and they have been applied successfully in a number of applications, including optimal control. (See, for example, Clarke [42], Dunn [77].) We describe a few of the relevant concepts here.

Given a constrained optimization problem formulated as

$$\min\ f(x)\ \text{ subject to } x \in \Omega, \tag{A.21}$$

where Ω is a closed subset of $\mathbb{R}^n$, most of the theory revolves around *tangent cones* and *normal cones* to the set Ω at various feasible points $x \in \Omega$. A definition of a tangent vector is given by Clarke [42, Theorem 2.4.5].

Definition A.1 (Tangent).

A vector $w \in \mathbb{R}^n$ is tangent *to Ω at $x \in \Omega$ if for all vector sequences $\{x_i\}$ with $x_i \to x$ and $x_i \in \Omega$, and all positive scalar sequences $t_i \downarrow 0$, there is a sequence $w_i \to w$ such that $x_i + t_i w_i \in \Omega$ for all i.*

The *tangent cone* $T_\Omega(x)$ is the collection of all tangent vectors to Ω at x. The *normal cone* $N_\Omega(x)$ is simply the orthogonal complement to the tangent cone; that is,

$$N_\Omega(x) = \left\{ v \mid v^T w \le 0 \text{ for all } w \in T_\Omega(x) \right\}. \tag{A.22}$$

Note that the zero vector belongs to both T_Ω and N_Ω.

It is not difficult to verify that both T_Ω and N_Ω are indeed cones according to the definition (A.2). To prove this for T_Ω, we take $w \in T_\Omega(x)$ and show that $\alpha w \in T_\Omega(x)$ for some given $\alpha \ge 0$. Let the sequence $\{x_i\}$ and $\{t_i\}$ be given, as in Definition A.1, and define the sequence $\{\bar{t}_i\}$ by $\bar{t}_i = t_i/\alpha$. Since $\{\bar{t}_i\}$ is also a valid sequence of scalars according to Definition A.1, there is a sequence of vectors $\{\bar{w}_i\}$ such that $x_i + \bar{t}_i \bar{w}_i \in \Omega$ and $\bar{w}_i \to w$. By defining $w_i = \alpha \bar{w}_i$, we have that $x_i + t_i w_i = x_i + \bar{t}_i \bar{w}_i \in \Omega$. Hence we have identified a sequence $\{w_i\}$ satisfying the conditions of Definition A.1 with the property that $w_i \to \alpha w$, so we conclude that $\alpha w \in T_\Omega(x)$, as required.

As an example, consider the set defined by a single equality constraint:

$$\Omega = \{x \in \mathbb{R}^n \mid h(x) = 0\}, \tag{A.23}$$

where $h : \mathbb{R}^n \to \mathbb{R}$ is smooth, and let x be a particular point for which $\nabla h(x) \ne 0$. If $w \in T_\Omega(x)$ for some $x \in \Omega$, we have from Definition A.1 that there is a sequence $w_i \to w$ such that $x + t_i w_i \in \Omega$ for any sequence $t_i \downarrow 0$. (We have made the legal choice $x_i \equiv x$ in the definition.) Therefore, we have

$$h(x + t_i w_i)\ \text{ for all } i, \qquad h(x) = 0.$$

By using this fact together with a Taylor series expansion and smoothness of h, we find that

$$0 = \frac{1}{t_i}\left[h(x + t_i w_i) - h(x)\right] = w_i^T \nabla h(x) + o(t_i)/t_i \to w^T \nabla h(x).$$

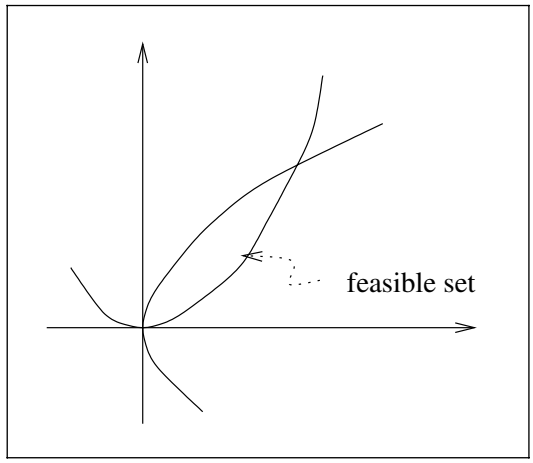


Figure A.1
Feasible set defined by $x_1 \geq x_2^2$, $x_2 \geq x_1^2$.

An argument based on the implicit function theorem can be used to prove the converse, which is that $w^T \nabla h(x) = 0 \Rightarrow w \in T_\Omega(x)$. Hence we have

$$T_\Omega(x) = \left\{ w \mid w^T \nabla h(x) = 0 \right\} = \text{Null}\left(\nabla h(x)^T \right). \tag{A.24}$$

It follows from the definition (A.22) that the normal cone is

$$N_\Omega(x) = \{ \alpha \nabla h(x) \mid \alpha \in \mathbb{R} \} = \text{Range}(\nabla h(x)). \tag{A.25}$$

For the next example, consider a set Ω defined by two parabolic constraints:

$$\Omega = \{ x \in \mathbb{R}^2 \mid x_1 \geq x_2^2,\ x_2 \geq x_1^2 \}$$

(see Figure A.1). The tangent and normal cones at the most interesting point—the origin—are given by

$$T_\Omega(0,0) = \{ w \in \mathbb{R}^2 \mid w_1 \geq 0,\, w_2 \geq 0 \},$$
$$N_\Omega(0,0) = \{ v \in \mathbb{R}^2 \mid v_1 \leq 0,\, v_2 \leq 0 \}.$$

To show that, for instance, the tangent vector $(0, 1)$ fits into Definition A.1, suppose we are given the feasible sequence $x_i = (1/i^2, 1/i) \to (0, 0)$ and the positive scalar sequence $t_i = 1/i \downarrow 0$. If we define $w_i = \left(\frac{1}{3i}, 1\right)$, it is easy to check that $x_i + t_i w_i$ is feasible for each i and that $w_i \to (0, 1)$.

A three-dimensional example is given by the "ice-cream cone" defined by

$$\Omega = \left\{ x \in \mathbb{R}^3 \mid x_3 \geq 2\sqrt{x_1^2 + x_2^2} \right\} \tag{A.26}$$

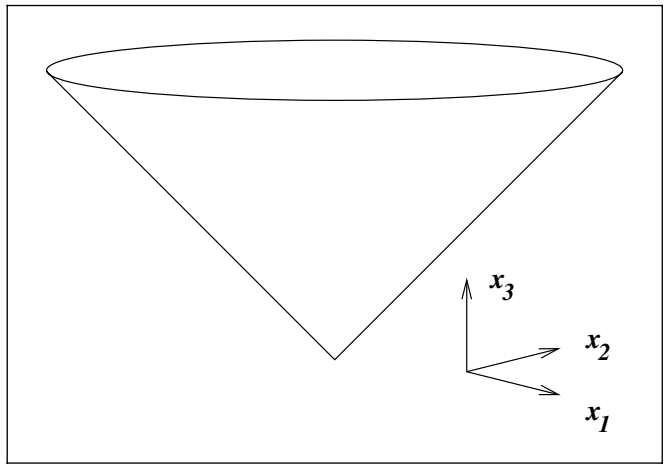


Figure A.2 "Ice-cream cone" set.

(see Figure A.2). At the origin $x = 0$, the tangent cone coincides with the feasible set: $T_\Omega(0) = \Omega$. The normal cone is defined as

$$N_\Omega(0) = \left\{ v \mid v_3 \leq -\frac{1}{2}\sqrt{v_1^2 + v_2^2} \right\}.$$

To verify that $v^T w \leq 0$ for all $v \in N_\Omega(0)$, $w \in T_\Omega(0)$, we have

$$\begin{aligned} v^T w &= v_1 w_1 + v_2 w_2 + v_3 w_3 \\ &\leq v_1 w_1 + v_2 w_2 - \frac{1}{2}(v_1^2 + v_2^2)^{1/2}(2)(w_1^2 + w_2^2)^{1/2} \\ &= \sqrt{(v_1 w_1 + v_2 w_2)^2} - \sqrt{(v_1^2 + v_2^2)(w_1^2 + w_2^2)} \\ &= \sqrt{v_1^2 w_1^2 + v_2^2 w_2^2 + 2 v_1 v_2 w_1 w_2} - \sqrt{v_1^2 w_1^2 + v_2^2 w_2^2 + v_2^2 w_1^2 + v_1^2 w_2^2}. \end{aligned}$$

Since

$$v_1^2 w_2^2 + v_2^2 w_1^2 \geq 2 v_1 v_2 w_2 w_2,$$

the second term outweighs the first, so we have $v^T w \leq 0$, as required.

A final example is given by the two-dimensional subset

$$\Omega = \left\{ x \in \mathbb{R}^2 \mid x_2 \geq 0, \quad x_2 \leq x_1^3 \right\}, \tag{A.27}$$

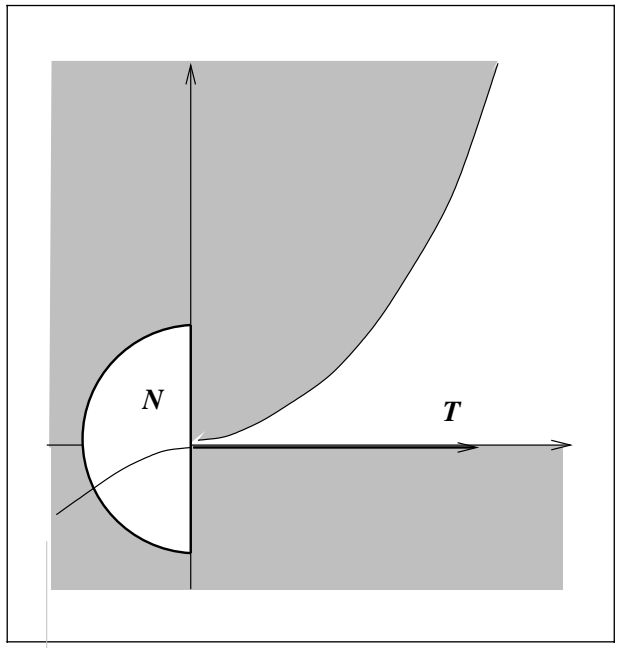


Figure A.3 Feasible set defined by $x_2 \geq 0$, $x_2 \leq x_1^3$, showing tangent and normal cones.

illustrated in Figure A.3. The tangent cone at the origin is

$$T_\Omega(0) = \{(w_1, 0) \mid w_1 \geq 0\}, \tag{A.28}$$

that is, all positive multiples of the vector $(1, 0)$. The normal cone is

$$N_\Omega(0) = \{v \mid v_1 \leq 0\}. \tag{A.29}$$

Finally, we recall the definitions of affine hull $\mathrm{aff}\,\mathcal{F}$ and relative interior $\mathrm{ri}\,\mathcal{F}$ from earlier in the chapter. For the set Ω defined by (A.27), we have that

$$\mathrm{ri}\Omega = \left\{x \mid x_2 > 0, \quad x_2 < x_1^3\right\},$$

$$\mathrm{ri}T_\Omega(0) = \{(w_1, 0) \mid w_1 > 0\},$$

$$\mathrm{ri}N_\Omega(0) = \{(v_1, v_2) \mid v_1 < 0\}.$$

ORDER NOTATION

In much of our analysis we are concerned with how the members of a sequence behave *eventually*, that is, when we get far enough along in the sequence. For instance, we might ask whether the elements of the sequence are bounded, or whether they are similar in size to the elements of a corresponding sequence, or whether they are decreasing and, if so, how rapidly. *Order notation* is useful shorthand to use when questions like these are being examined. It saves us defining many constants that clutter up the argument and the analysis.

We will use three varieties of order notation: $O(\cdot)$, $o(\cdot)$, and $\Omega(\cdot)$. Given two nonnegative infinite sequences of scalars $\{\eta_k\}$ and $\{\nu_k\}$, we write

$$\eta_k = O(\nu_k)$$

if there is a positive constant C such that

$$|\eta_k| \leq C|\nu_k|$$

for all k sufficiently large. We write

$$\eta_k = o(\nu_k)$$

if the sequence of ratios $\{\eta_k/\nu_k\}$ approaches zero, that is,

$$\lim_{k\to\infty} \frac{\eta_k}{\nu_k} = 0.$$

Finally, we write

$$\eta_k = \Omega(\nu_k)$$

if there are two constants C_0 and C_1 with $0 < C_0 \leq C_1 < \infty$ such that

$$C_0|\nu_k| \leq |\eta_k| \leq C_1|\nu_k|,$$

that is, the corresponding elements of both sequences stay in the same ballpark for all k. This definition is equivalent to saying that $\eta_k = O(\nu_k)$ and $\nu_k = O(\eta_k)$.

The same notation is often used in the context of quantities that depend continuously on each other as well. For instance, if $\eta(\cdot)$ is a function that maps $\mathbb{R}$ to $\mathbb{R}$, we write

$$\eta(\nu) = O(\nu)$$

if there is a constant C such that $|\eta(\nu)| \leq C|\nu|$ for all $\nu \in \mathbb{R}$. (Typically, we are interested only in values of ν that are either very large or very close to zero; this should be clear from

the context. Similarly, we use

$$\eta(\nu) = o(\nu) \tag{A.30}$$

to indicate that the ratio $\eta(\nu)/\nu$ approaches zero either as $\nu \to 0$ or $\nu \to \infty$. (Again, the precise meaning should be clear from the context.)

As a slight variant on the definitions above, we write

$$\eta_k = O(1)$$

to indicate that there is a constant C such that $|\eta_k| \leq C$ for all k, while

$$\eta_k = o(1)$$

indicates that $\lim_{k\to\infty} \eta_k = 0$. We sometimes use vector and matrix quantities as arguments, and in these cases the definitions above are intended to apply to the norms of these quantities. For instance, if $f : \mathbb{R}^n \to \mathbb{R}^n$, we write $f(x) = O(\|x\|)$ if there is a constant $C > 0$ such that $\|f(x)\| \leq C\|x\|$ for all x in the domain of f. Typically, as above, we are interested only in some subdomain of f, usually a small neighborhood of 0. As before, the precise meaning should be clear from the context.

ROOT-FINDING FOR SCALAR EQUATIONS

In Chapter 11 we discussed methods for finding solutions of nonlinear systems of equations $F(x) = 0$, where $F : \mathbb{R}^n \to \mathbb{R}^n$. Here we discuss briefly the case of scalar equations ($n = 1$), for which the algorithm is easy to illustrate. Scalar root-finding is needed in the trust-region algorithms of Chapter 4, for instance. Of course, the general theorems of Chapter 11 can be applied to derive rigorous convergence results for this special case.

The basic step of Newton's method (Algorithm Newton of Chapter 11) in the scalar case is simply

$$p_k = -F(x_k)/F'(x_k), \qquad x_{k+1} \leftarrow x_k + p_k \tag{A.31}$$

(cf. (11.9)). Graphically, such a step involves taking the tangent to the graph of F at the point x_k and taking the next iterate to be the intersection of this tangent with the x axis (see Figure A.4). Clearly, if the function F is nearly linear, the tangent will be quite a good approximation to F itself, so the Newton iterate will be quite close to the true root of F.

The secant method for scalar equations can be viewed as the specialization of Broyden's method to the case of $n = 1$. The issues are simpler in this case, however, since the secant equation (11.26) completely determines the value of the 1×1 approximate Hessian B_k. That is, we do not need to apply extra conditions to ensure that B_k is fully determined. By

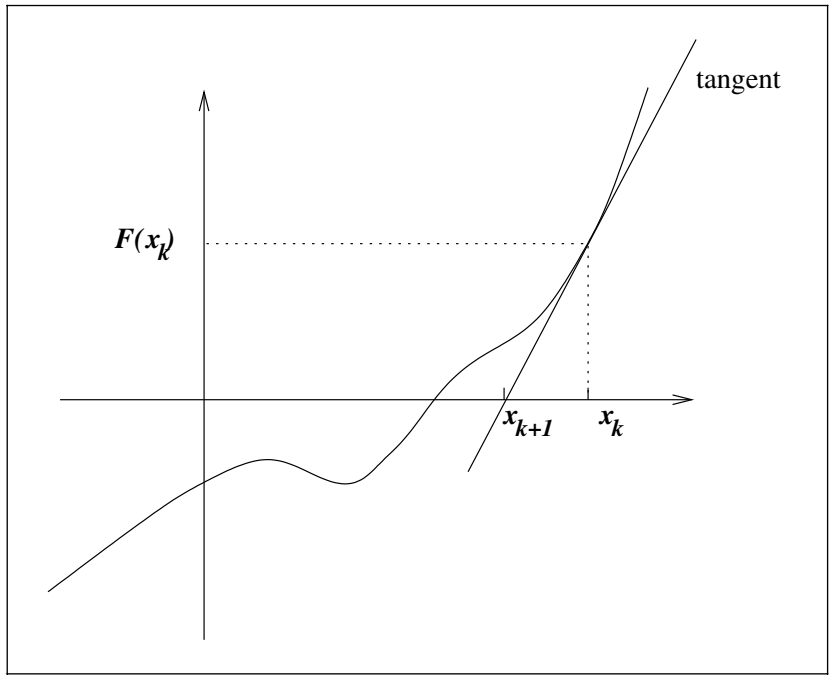


Figure A.4 One step of Newton's method for a scalar equation.

combining (11.24) with (11.26), we find that the secant method for the case of $n = 1$ is defined by

$$B_k = (F(x_k) - F(x_{k-1}))/(x_k - x_{k-1}), \tag{A.32a}$$

$$p_k = -F(x_k)/B_k, \qquad x_{k+1} = x_k + p_k. \tag{A.32b}$$

By illustrating this algorithm, we see the origin of the term "secant." B_k approximates the slope of the function at x_k by taking the secant through the points $(x_{k-1}, F(x_{k-1})$ and $(x_k, F(x_k))$, and x_{k+1} is obtained by finding the intersection of this secant with the x axis. The method is illustrated in Figure A.5.

A.2 ELEMENTS OF LINEAR ALGEBRA

VECTORS AND MATRICES

In this book we work exclusively with vectors and matrices whose components are real numbers. Vectors are usually denoted by lowercase roman characters, and matrices by

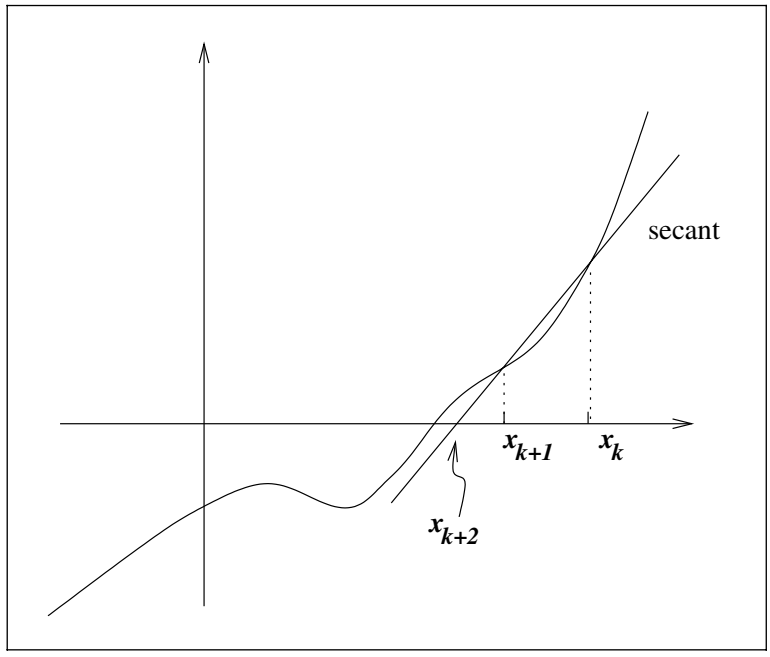


Figure A.5 One step of the secant method for a scalar equation.

uppercase roman characters. The space of real vectors of length n is denoted by $\mathbb{R}^n$, while the space of real $m \times n$ matrices is denoted by $\mathbb{R}^{m\times n}$.

We say that a matrix $A \in \mathbb{R}^{n\times n}$ is *symmetric* if $A = A^T$. A symmetric matrix A is *positive definite* if there is a positive constant α such that

$$x^T Ax \geq \alpha \|x\|^2, \qquad \text{for all } x \in \mathbb{R}^n. \tag{A.33}$$

It is *positive semidefinite* if the relationship (A.33) holds with $\alpha = 0$, that is, $x^T Ax \geq 0$ for all $x \in \mathbb{R}^n$.

NORMS

For a vector $x \in \mathbb{R}^n$, we define the following norms:

$$\|x\|_1 \stackrel{\text{def}}{=} \sum_{i=1}^{n} |x_i|, \tag{A.34a}$$

$$\|x\|_2 \stackrel{\text{def}}{=} \left(\sum_{i=1}^{n} x_i^2 \right)^{1/2} = (x^T x)^{1/2}, \tag{A.34b}$$

$$\|x\|_\infty \stackrel{\text{def}}{=} \max_{i=1,\ldots,n} |x_i|. \tag{A.34c}$$

The norm $\|\cdot\|_2$ is often called the *Euclidean norm*. All these norms measure the length of the vector in some sense, and they are equivalent in the sense that each one is bounded above and below by a multiple of the other. To be precise, we have

$$\|x\|_\infty \le \|x\|_2 \le \sqrt{n}\|x\|_\infty, \qquad \|x\|_\infty \le \|x\|_1 \le n\|x\|_\infty, \tag{A.35}$$

and so on. In general, a norm is any mapping $\|\cdot\|$ from $\mathbb{R}^n$ to the nonnegative real numbers that satisfies the following properties:

$$\|x+z\| \le \|x\| + \|z\|, \qquad \text{for all } x, z \in \mathbb{R}^n; \tag{A.36a}$$

$$\|x\| = 0 \Rightarrow x = 0; \tag{A.36b}$$

$$\|\alpha x\| = |\alpha|\|x\|, \qquad \text{for all } \alpha \in \mathbb{R} \text{ and } x \in \mathbb{R}^n. \tag{A.36c}$$

Equality holds in (A.36a) if and only if one of the vectors x and z is a nonnegative scalar multiple of the other.

Another interesting property that holds for the Euclidean norm $\|\cdot\| = \|\cdot\|_2$ is the Hölder inequality, which states that

$$\left|x^T z\right| \le \|x\|\,\|z\|, \tag{A.37}$$

with equality if and only if one of these vectors is a nonnegative multiple of the other. We can prove this result as follows:

$$0 \le \|\alpha x + z\|^2 = \alpha^2\|x\|^2 + 2\alpha x^T z + \|z\|^2.$$

The right-hand-side is a convex function of α, and it satisfies the required nonnegativity property only if there exist fewer than 2 distinct real roots, that is,

$$(2x^T z)^2 \le 4\|x\|^2\|z\|^2,$$

proving (A.37). Equality occurs when the quadratic α has exactly one real root (that is, $|x^T z| = \|x\|\,\|z\|$) and when $\alpha x + z = 0$ for some α, as claimed.

We can derive definitions for certain matrix norms from these vector norm definitions. If we let $\|\cdot\|$ be generic notation for the three norms listed in (A.34), we define the corresponding matrix norm as

$$\|A\| \stackrel{\text{def}}{=} \sup_{x \neq 0} \frac{\|Ax\|}{\|x\|}. \tag{A.38}$$

The matrix norms defined in this way are said to be *consistent* with the vector norms (A.34). Explicit formulae for these norms are as follows:

$$\|A\|_1 = \max_{j=1,\ldots,n} \sum_{i=1}^{m} |A_{ij}|, \tag{A.39a}$$

$$\|A\|_2 = \lambda_1(A^T A)^{1/2}, \quad \text{where } \lambda_1(\cdot) \text{ denotes the largest eigenvalue,} \tag{A.39b}$$

$$\|A\|_\infty = \max_{i=1,\ldots,m} \sum_{j=1}^{n} |A_{ij}|. \tag{A.39c}$$

The Frobenius norm $\|A\|_F$ of the matrix A is defined by

$$\|A\|_F = \left(\sum_{i=1}^{m} \sum_{j=1}^{n} a_{ij}^2 \right)^{1/2}. \tag{A.40}$$

This norm is useful for many purposes, but it is not consistent with any vector norm. Once again, these various matrix norms are equivalent with each other in a sense similar to (A.35).

For the Euclidean norm $\|\cdot\| = \|\cdot\|_2$, the following property holds:

$$\|AB\| \le \|A\| \, \|B\|, \tag{A.41}$$

for all matrices A and B with consistent dimensions.

The *condition number* of a nonsingular matrix is defined as

$$\kappa(A) = \|A\| \, \|A^{-1}\|, \tag{A.42}$$

where any matrix norm can be used in the definition. We distinguish the different norms by the use of a subscript: $\kappa_1(\cdot)$, $\kappa_2(\cdot)$, and $\kappa_\infty(\cdot)$, respectively. (Of course, different norm definitions give different measures of condition number, in general.)

Norms also have a meaning for scalar, vector, and matrix-valued functions that are defined on a particular domain. In these cases, we can define Hilbert spaces of functions for which the inner product and norm are defined in terms of an integral over the domain. We omit details, since all the development of this book takes place in the space $\mathbb{R}^n$, though many of the algorithms can be extended to more general Hilbert spaces. However, we mention for purposes of Newton analysis that the following inequality holds for functions of the type that we consider in this book:

$$\left\| \int_a^b F(t) \right\| \le \int_a^b \|F(t)\| \, dt,$$

where F is a scalar-, vector-, or matrix-valued function on the interval $[a, b]$.

SUBSPACES

Given the Euclidean space $\mathbb{R}^n$, the subset $\mathcal{S} \subset \mathbb{R}^n$ is a *subspace of* $\mathbb{R}^n$ if the following property holds: If x and y are any two elements of $\mathcal{S}$, then

$$\alpha x + \beta y \in \mathcal{S}, \quad \text{for all } \alpha, \beta \in \mathbb{R}.$$

For instance, $\mathcal{S}$ is a subspace of $\mathbb{R}^2$ if it consists of (i) the whole space $\mathbb{R}^n$; (ii) any line passing through the origin; (iii) the origin alone; or (iv) the empty set.

Given any set of vectors $a_i \in \mathbb{R}^n$, $i = 1, 2, \ldots, m$, the set

$$\mathcal{S} = \left\{ w \in \mathbb{R}^n \mid a_i^T w = 0,\ i = 1, 2, \ldots, m \right\} \tag{A.43}$$

is a subspace. However, the set

$$\left\{ w \in \mathbb{R}^n \mid a_i^T w \geq 0,\ i = 1, 2, \ldots, m \right\} \tag{A.44}$$

is not in general a subspace. For example, if we have $n = 2, m = 1$, and $a_1 = (1, 0)^T$, this set would consist of all vectors $(w_1, w_2)^T$ with $w_1 \geq 0$, but then given two vectors $x = (1, 0)^T$ and $y = (2, 3)$ in this set, it is easy to choose multiples α and β such that $\alpha x + \beta y$ has a negative first component, and so lies outside the set.

Sets of the forms (A.43) and (A.44) arise in the discussion of second-order optimality conditions for constrained optimization.

For any subspace $\mathcal{S}$ of the Euclidean space $\mathbb{R}^n$, the set of vectors $s_1, s_2, \ldots, s_m$ in $\mathcal{S}$ is called a *linearly independent set* if there are no real numbers $\alpha_1, \alpha_2, \ldots, \alpha_m$ such that

$$\alpha_1 s_2 + \alpha_2 s_2 + \cdots + \alpha_m s_m,$$

unless we make the trivial choice $\alpha_1 = \alpha_2 = \cdots = \alpha_m = 0$. Another way to define linear independence is to say that none of the vectors $s_1, s_2, \ldots, s_m$ can be written as a linear combination of the other vectors in this set. We say that this set of vectors is a *spanning set* for $\mathcal{S}$ if *any* vector $s \in \mathcal{S}$ can be written as

$$s = \alpha_1 s_2 + \alpha_2 s_2 + \cdots + \alpha_m s_m,$$

for some particular choice of the coefficients $\alpha_1, \alpha_2, \ldots, \alpha_m$.

If the vectors $s_1, s_2, \ldots, s_m$ are both linearly independent and a spanning set for $\mathcal{S}$, we call them a *basis*. In this case, m (the number of elements in the basis) is referred to as the *dimension* of $\mathcal{S}$. Notationally, we write $\dim(\mathcal{S})$ to denote the dimension of $\mathcal{S}$. Note that there are many ways to choose a basis of $\mathcal{S}$ in general, but that all bases do, in fact, contain the same number of vectors.

If A is any real matrix, the *null space* is the subspace

$$\text{Null}(A) = \{w \mid Aw = 0\},$$

while the *range space* is

$$\text{Range}(A) = \{w \mid w = Av \text{ for some vector } v\}.$$

The *fundamental theorem of linear algebra* states that

$$\text{Null}(A) \oplus \text{Range}(A^T) = \mathbb{R}^n,$$

where n is the number of columns in A.

EIGENVALUES, EIGENVECTORS, AND THE SINGULAR-VALUE DECOMPOSITION

A scalar value λ is an *eigenvalue* of the $n \times n$ matrix A if there is a nonzero vector x such that

$$Aq = \lambda q.$$

The vector q is called an *eigenvector* of A. The matrix A is nonsingular if none of its eigenvalues are zero. The eigenvalues of symmetric matrices are all real numbers, while non symmetric matrices may have imaginary eigenvalues. If the matrix is positive definite as well as symmetric, its eigenvalues are all positive real numbers.

All matrices $A \in \mathbb{R}^{m \times n}$ can be decomposed as a product of three matrices with special properties, as follows:

$$A = USV^T. \tag{A.45}$$

Here, U and V are orthogonal matrices of dimension $m \times m$ and $n \times n$, respectively, which means that they satisfy the relations $U^TU = UU^T = I$ and $V^TV = VV^T = I$. The matrix S is a diagonal matrix of dimension $m \times n$, wih diagonal elements $\sigma_i, i = 1, 2, \ldots, \min(m, n)$, that satisfy

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{\min(m,n)} \geq 0.$$

These diagonal values are called the singular values of A, and (A.45) is called the singular-value decomposition.

When A is symmetric, its n real eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$ and their associated eigenvectors $q_1, q_2, \ldots, q_n$ can be used to write a *spectral decomposition* of A as follows:

$$A = \sum_{i=1}^{n} \lambda_i q_i q_i^T.$$

This decomposition can be restated in matrix form by defining

$$\Lambda = \text{diag}(\lambda_1, \lambda_2, \cdots, \lambda_n), \qquad Q = [q_1 \,|\, q_2 \,|\, \ldots \,|\, q_n],$$

and writing

$$A = Q \Lambda Q^T. \tag{A.46}$$

In fact, when A is positive definite as well as symmetric, this decomposition is identical to the singular-value decomposition (A.45), where we define $U = V = Q$ and $S = \Lambda$. Note that the singular values σ_i, $i = 1, 2, \ldots, m$, and the eigenvalues λ_i, $i = 1, 2, \ldots, m$, coincide in this case.

In the case of the Euclidean norm (A.39b), we have for symmetric positive definite matrices A that the singular values and eigenvalues of A coincide, and that

$$\begin{aligned} \|A\| &= \sigma_1(A) = \text{largest eigenvalue of } A, \\ \|A^{-1}\| &= \sigma_n(A)^{-1} = \text{inverse of smallest eigenvalue of } A. \end{aligned}$$

Hence, we have for all $x \in \mathbb{R}^n$ that

$$\sigma_n(A)\|x\|^2 = \|x\|^2/\|A^{-1}\| \leq x^T A x \leq \|A\|\|x\|^2 = \sigma_1(A)\|x\|^2.$$

For an orthogonal matrix Q, we have for the Euclidean norm that

$$\|Qx\| = \|x\|,$$

and that all the singular values of this matrix are equal to 1.

DETERMINANT AND TRACE

The trace of an $n \times n$ matrix A is defined by

$$\text{trace}(A) = \sum_{i=1}^{n} A_{ii}. \tag{A.47}$$

If the eigenvalues of A are denoted by $\lambda_1, \lambda_2, \ldots, \lambda_n$, it can be shown that

$$\text{trace}(A) = \sum_{i=1}^{n} \lambda_i, \tag{A.48}$$

that is, the trace of the matrix is the sum of its eigenvalues.

The determinant of an $n \times n$ matrix A is the product of its eigenvalues; that is,

$$\det A = \prod_{i=1}^{n} \lambda_i. \tag{A.49}$$

The determinant has several appealing (and revealing) properties. For instance,

$\det A = 0$ if and only if A is singular;

$\det AB = (\det A)(\det B)$;

$\det A^{-1} = 1/\det A$.

Recall that any orthogonal matrix A has the property that $QQ^T = Q^TQ = I$, so that $Q^{-1} = Q^T$. It follows from the property of the determinant that $\det Q = \det Q^T = \pm 1$.

The properties above are used in the analysis of Chapters 6 and 8.

MATRIX FACTORIZATIONS: CHOLESKY, LU, QR

Matrix factorizations are important both in the design of algorithms and in their analysis. One such factorization is the singular-value decomposition defined above in (A.45). Here we define the other important factorizations.

All the factorization algorithms described below make use of *permutation matrices*. Suppose that we wish to exchange the first and fourth rows of a matrix A. We can perform this operation by premultiplying A by a permutation matrix P, which is constructed by interchanging the first and fourth rows of an identity matrix that contains the same number of rows as A. Suppose, for example, that A is a 5×5 matrix. The appropriate choice of P would be

$$P = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

A similar technique is used to to find a permutation matrix P that exchanges columns of a matrix.

The LU factorization of a matrix $A \in \mathbb{R}^{n\times n}$ is defined as

$$PA = LU, \tag{A.50}$$

where

P is an $n \times n$ permutation matrix (that is, it is obtained by rearranging the rows of the $n \times n$ identity matrix),

L is unit lower triangular (that is, lower triangular with diagonal elements equal to 1, and

U is upper triangular.

This factorization can be used to solve a linear system of the form $Ax = b$ efficiently by the following three-step process:

form $\tilde{b} = Pb$ by permuting the elements of b;

solve $Lz = \tilde{b}$ by performing triangular forward-substitution, to obtain the vector z;

solve $Ux = z$ by performing triangular back-substitution, to obtain the solution vector x.

The factorization (A.50) can be found by using Gaussian elimination with row partial pivoting, an algorithm that requires approximately $2n^3/3$ floating-point operations when A is dense. Standard software that implements this algorithm (notably, LAPACK [4]) is readily available. The method can be stated as follows:

Algorithm A.1 (Gaussian Elimination with Row Partial Pivoting).
Given $A \in \mathbb{R}^{n\times n}$;
Set $P \leftarrow I, L \leftarrow 0$;
for $i = 1, 2, \ldots, n$
    find the index $j \in \{i, i+1, \ldots, n\}$ such that $|A_{ji}| = \max_{k=i,i+1,\ldots,n} |A_{ki}|$;
    if $A_{ij} = 0$
        stop; (* matrix A is singular *)
    if $i \neq j$
        swap rows i and j of matrices A and L;
    (* elimination step*)
    $L_{ii} \leftarrow 1$;
    for $k = i+1, i+2, \ldots, n$
        $L_{ki} \leftarrow A_{ki}/A_{ii}$;
        for $l = i+1, i+2, \ldots, n$
            $A_{kl} \leftarrow A_{kl} - L_{ki}A_{il}$;
        end (for)
    end (if)

end (for)
$U \leftarrow$ upper triangular part of A.

Variants of the basic algorithm allow for rearrangement of the columns as well as the rows during the factorization, but these do not add to the practical stability properties of the algorithm. Column pivoting may, however, improve the performance of Gaussian elimination when the matrix A is sparse, by ensuring that the factors L and U are also reasonably sparse.

Gaussian elimination can be applied also to the case in which A is not square. When A is $m \times n$, with $m > n$, the standard row pivoting algorithm produces a factorization of the form (A.50), where $L \in \mathbb{R}^{m \times n}$ is unit lower triangular and $U \in \mathbb{R}^{n \times n}$ is upper triangular. When $m < n$, we can find an LU factorization of A^T rather than A, that is, we obtain

$$PA^T = \begin{bmatrix} L_1 \\ L_2 \end{bmatrix} U, \tag{A.51}$$

where L_1 is $m \times m$ (square) unit lower triangular, U is $m \times m$ upper triangular, and L_2 is a general $(n-m) \times m$ matrix. If A has full row rank, we can use this factorization to calculate its null space explicitly as the space spanned by the columns of the matrix

$$M = P^T \begin{bmatrix} L_1^{-T} L_2^T \\ -I \end{bmatrix} U^{-T}. \tag{A.52}$$

(It is easy to check that M has dimensions $n \times (n-m)$ and that $AM = 0$; we leave this an exercise.)

When $A \in \mathbb{R}^{n \times n}$ is symmetric positive definite, it is possible to compute a similar but more specialized factorization at about half the cost—about $n^3/3$ operations. This factorization, known as the Cholesky factorization, produces a matrix L such that

$$A = LL^T. \tag{A.53}$$

(If we require L to have positive diagonal elements, it is uniquely defined by this formula.) The algorithm can be specified as follows.

Algorithm A.2 (Cholesky Factorization).
Given $A \in \mathbb{R}^{n \times n}$ symmetric positive definite;
for $i = 1, 2, \ldots, n$;
    $L_{ii} \leftarrow \sqrt{A_{ii}}$;
    for $j = i+1, i+2, \ldots, n$
        $L_{ji} \leftarrow A_{ji}/L_{ii}$;
        for $k = i+1, i+2, \ldots, j$
            $A_{jk} \leftarrow A_{jk} - L_{ji}L_{ki}$;

```
            end (for)
        end (for)
    end (for)
```

Note that this algorithm references only the lower triangular elements of A; in fact, it is only necessary to store these elements in any case, since by symmetry they are simply duplicated in the upper triangular positions.

Unlike the case of Gaussian elimination, the Cholesky algorithm can produce a valid factorization of a symmetric positive definite matrix without swapping any rows or columns. However, symmetric permutation (that is, reordering the rows and columns in the same way) can be used to improve the sparsity of the factor L. In this case, the algorithm produces a permutation of the form

$$P^T A P = L L^T$$

for some permutation matrix P.

The Cholesky factorization can be used to compute solutions of the system $Ax = b$ by performing triangular forward- and back-substitutions with L and L^T, respectively, as in the case of L and U factors produced by Gaussian elimination.

Another useful factorization of rectangular matrices $A \in \mathbb{R}^{m \times n}$ has the form

$$AP = QR, \tag{A.54}$$

where

P is an $n \times n$ permutation matrix,

A is $m \times m$ orthogonal, and

R is $m \times n$ upper triangular.

In the case of a square matrix $m = n$, this factorization can be used to compute solutions of linear systems of the form $Ax = b$ via the following procedure:

set $\tilde{b} = Q^T b$;

solve $Rz = \tilde{b}$ for z by performing back-substitution;

set $x = P^T z$ by rearranging the elements of x.

For a dense matrix A, the cost of computing the QR factorization is about $4m^2 n/3$ operations. In the case of a square matrix, this is about twice as many operations as required to compute an LU factorization via Gaussian elimination. Unlike the case of Gaussian elimination, the QR factorization procedure cannot be modified in general to ensure efficiency on sparse matrices. That is, no matter how the column permutation matrix P is chosen, the factors Q

and R will be dense in general. (This remains true even if we allow row pivoting as well as column pivoting.)

Algorithms to perform QR factorization are almost as simple as algorithms for Gaussian elimination and for Cholesky factorization. The most widely used algorithms work by applying a sequence of special orthogonal matrices to A, known either as Householder transformations or Givens rotations, depending on the algorithm. We omit the details, and refer instead to Golub and Van Loan [115, Chapter 5] for a complete description.

In the case of a rectangular matrix A with $m < n$, we can use the QR factorization of A^T to find a matrix whose columns span the null space of A. To be specific, we write

$$A^T P = QR = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} R,$$

where Q_1 consists of the first m columns of Q, and Q_2 contains the last $n - m$ columns. It is easy to show that columns of the matrix Q_2 span the null space of A. This procedure yields a more satisfactory basis matrix for the null space than the Gaussian elimination procedure (A.52), because the columns of Q_2 are orthogonal to each other and have unit length. It may be more expensive to compute, however, particularly in the case in which A is sparse.

When A has full column rank, we can make an identification between the R factor in (A.54) and the Cholesky factorization. By multiplying the formula (A.54) by its transpose, we obtain

$$P^T A^T A P = R^T Q^T Q R = R^T R,$$

and by comparison with (A.53), we see that R^T is simply the Cholesky factor of the symmetric positive definite matrix $P^T A^T A P$. Recalling that L is uniquely defined when we restrict its diagonal elements to be positive, this observation implies that R is also uniquely defined for a given choice of permutation matrix P, provided that we enforce positiveness of the diagonals of R. Note, too, that since we can rearrange (A.54) to read $APR^{-1} = Q$, we can conclude that Q is also uniquely defined under these conditions.

Note that by definition of the Euclidean norm and the property (A.41), and the fact that the Euclidean norms of the matrices P and Q in (A.54) are both 1, we have that

$$\|A\| = \|QRP^T\| \leq \|Q\| \, \|R\| \, \|P^T\| = \|R\|,$$

while

$$\|R\| = \|Q^T A P\| \leq \|Q^T\| \, \|A\| \, \|P\| = \|A\|.$$

We conclude from these two inequalities that $\|A\| = \|R\|$. When A is square, we have by a similar argument that $\|A^{-1}\| = \|R^{-1}\|$. Hence the Euclidean-norm condition number of A can be estimated by substituting R for A in the expression (A.42). This observation is

significant because various techniques are available for estimating the condition number of triangular matrices R; see Golub and Van Loan [115, pp. 128–130] for a discussion.

SHERMAN–MORRISON–WOODBURY FORMULA

If the square nonsingular matrix A undergoes a rank-one update to become

$$\bar{A} = A + ab^T,$$

where $a, b \in \mathbb{R}^n$, then if $\bar{A}$ is nonsingular, we have

$$\bar{A}^{-1} = A^{-1} - \frac{A^{-1}ab^TA^{-1}}{1 + b^TA^{-1}a}. \tag{A.55}$$

It is easy to verify this formula: Simply multiply the definitions of $\bar{A}$ and $\bar{A}^{-1}$ together and check that they produce the identity.

This formula can be extended to higher-rank updates. Let U and V be matrices in $\mathbb{R}^{n\times p}$ for some p between 1 and n. If we define

$$\hat{A} = A + UV^T,$$

then

$$\hat{A}^{-1} = A^{-1} - A^{-1}U(I + V^TA^{-1}U)^{-1}V^TA^{-1}. \tag{A.56}$$

We can use this formula to solve linear systems of the form $\bar{A}x = d$. Since

$$x = \hat{A}^{-1}d = A^{-1}d - A^{-1}U(I + V^TA^{-1}U)^{-1}V^TA^{-1}d,$$

we see that x can be found by solving $p+1$ linear systems with the matrix A (to obtain $A^{-1}d$ and $A^{-1}U$), inverting the $p \times p$ matrix $I + V^TA^{-1}U$, and performing some elementary matrix algebra. Inversion of the $p \times p$ matrix $I + V^TA^{-1}U$ is inexpensive when $p \ll n$.

INTERLACING EIGENVALUE THEOREM

The following result is proved in Golub and Van Loan [115, Theorem 8.1.8].

Theorem A.2 (Interlacing Eigenvalue Theorem).
Let $A \in \mathbb{R}^{n\times n}$ be a symmetric matrix with eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$ satisfying

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n,$$

and let $z \in \mathbb{R}^n$ be a vector with $\|z\| = 1$, and $\alpha \in \mathbb{R}$ be a scalar. Then if we denote the eigenvalues of $A + \alpha zz^T$ by $\xi_1, \xi_2, \ldots, \xi_n$ (in decreasing order), we have for $\alpha > 0$ that

$$\xi_1 \geq \lambda_1 \geq \xi_2 \geq \lambda_2 \geq \xi_3 \geq \cdots \geq \xi_n \geq \lambda_n,$$

with

$$\sum_{i=1}^{n} \xi_i - \lambda_i = \alpha. \tag{A.57}$$

If $\alpha < 0$, we have that

$$\lambda_1 \geq \xi_1 \geq \lambda_2 \geq \xi_2 \geq \lambda_3 \geq \cdots \geq \lambda_n \geq \xi_n,$$

where the relationship (A.57) is again satisfied.

Informally stated, the eigenvalues of the modified matrix "interlace" the eigenvalues of the original matrix, with nonnegative adjustments if the coefficient α is positive, and nonpositive adjustments if α is negative. The total magnitude of the adjustments equals α, whose magnitude is identical to the Euclidean norm $\|\alpha zz^T\|_2$ of the modification.

ERROR ANALYSIS AND FLOATING-POINT ARITHMETIC

In most of this book our algorithms and analysis deal with real numbers. Modern digital computers, however, cannot store or compute with general real numbers. Instead, they work with a subset known as *floating-point numbers*. Any quantities that are stored on the computer, whether they are read directly from a file or program or arise as the intermediate result of a computation, must be approximated by a floating-point number. In general, then, the numbers that are produced by practical computation differ from those that would be produced if the arithmetic were exact. Of course, we try to perform our computations in such a way that these differences are as tiny as possible.

Discussion of errors requires us to distinguish between *absolute error* and *relative error*. If x is some exact quantity (scalar, vector, matrix) and $\tilde{x}$ is its approximate value, the absolute error is the norm of the difference, namely, $\|x - \tilde{x}\|$. (In general, any of the norms (A.34a), (A.34b), and (A.34c) can be used in this definition.) The relative error is the ratio of the absolute error to the size of the exact quantity, that is,

$$\frac{\|x - \tilde{x}\|}{\|x\|}.$$

When this ratio is significantly less than one, we can replace the denominator by the size of the approximate quantity—that is, $\|\tilde{x}\|$—without affecting its value very much.

Most computations associated with optimization algorithms are performed in double-precision arithmetic. Double-precision numbers are stored in words of length 64 bits. Most of these bits (say t) are devoted to storing the *fractional part*, while the remainder encode the *exponent* e and other information, such as the sign of the number, or an indication of whether it is zero or "undefined." Typically, the fractional part has the form

$$.d_1 d_2 \ldots d_t,$$

where each $d_i, i = 1, 2, \ldots, t$, is either zero or one. (In some systems d_1 is implicitly assumed to be 1 and is not stored.) The value of the floating-point number is then

$$\sum_{i=1}^{t} d_i 2^{-i} \times 2^e.$$

The value 2^{-t} is known as *unit roundoff* and is denoted by $\mathbf{u}$. Any real number whose absolute value lies in the range $[2^L, 2^U]$ (where L and U are lower and upper bounds on the value of the exponent e) can be approximated to within a relative accuracy of $\mathbf{u}$ by a floating-point number, that is,

$$\mathrm{fl}(x) = x(1 + \epsilon), \qquad \text{where } |\epsilon| \leq \mathbf{u}, \tag{A.58}$$

where $\mathrm{fl}(\cdot)$ denotes floating-point approximation. The value of $\mathbf{u}$ for double-precision computations is typically about 10^{-15}. In other words, if the real number x and its floating-point approximation are both written as base-10 numbers (the usual fashion), they agree to at least 15 digits.

For further information on floating-point computations, see Golub and Van Loan [115, Section 2.4] and Higham [136].

When an arithmetic operation is performed with one or two floating-point numbers, the result must also be stored as a floating-point number. This process introduces a small *roundoff error*, whose size can be quantified in terms of the size of the arguments. If x and y are two floating-point numbers, we have that

$$|\mathrm{fl}(x * y) - x * y| \leq \mathbf{u}|x * y|, \tag{A.59}$$

where $*$ denotes any of the operations $+, -, \times, \div$.

Although the error in a single floating-point operation appears benign, more significant errors may occur when the arguments x and y are floating-point approximations of two *real* numbers, or when a sequence of computations are performed in succession. Suppose, for instance, that x and y are large real numbers whose values are very similar. When we store them in a computer, we approximate them with floating-point numbers $\mathrm{fl}(x)$ and $\mathrm{fl}(y)$

that satisfy

$$\text{fl}(x) = x + \epsilon_x, \quad \text{fl}(y) = y + \epsilon_y, \quad \text{where } |\epsilon_x| \leq \mathbf{u}|x|, |\epsilon_y| \leq \mathbf{u}|y|.$$

If we take the difference of the two stored numbers, we obtain a final result $\text{fl}(\text{fl}(x) - \text{fl}(y))$ that satisfies

$$\text{fl}(\text{fl}(x) - \text{fl}(y)) = (\text{fl}(x) - \text{fl}(y))(1 + \epsilon_{xy}), \quad \text{where } |\epsilon_{xy}| \leq \mathbf{u}.$$

By combining these expressions, we find that the difference between this result and the true value $x - y$ may be as large as

$$\epsilon_x + \epsilon_y + \epsilon_{xy},$$

which is bounded by $\mathbf{u}(|x| + |y| + |x - y|)$. Hence, since x and y are large and close together, the relative error is approximately $2\mathbf{u}|x|/|x - y|$, which may be quite large, since $|x| \gg |x - y|$.

This phenomenon is known as *cancellation*. It can also be explained (less formally) by noting that if both x and y are accurate to k digits, and if they agree in the first $\bar{k}$ digits, then their difference will contain only about $k - \bar{k}$ significant digits—the first $\bar{k}$ digits cancel each other out. This observation is the reason for the well-known adage of numerical computing—that one should avoid taking the difference of two similar numbers if at all possible.

CONDITIONING AND STABILITY

Conditioning and *stability* are two terms that are used frequently in connection with numerical computations. Unfortunately, their meaning sometimes varies from author to author, but the general definitions below are widely accepted, and we adhere to them in this book.

Conditioning is a property of the numerical problem at hand (whether it is a linear algebra problem, an optimization problem, a differential equations problem, or whatever). A problem is said to be *well conditioned* if its solution is not affected greatly by small perturbations to the data that define the problem. Otherwise, it is said to be *ill conditioned*.

A simple example is given by the following 2×2 system of linear equations:

$$\begin{bmatrix} 1 & 2 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 3 \\ 2 \end{bmatrix}.$$

By computing the inverse of the coefficient matrix, we find that the solution is simply

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} -1 & 2 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 3 \\ 2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

If we replace the first right-hand-side element by 3.00001, the solution becomes $(x_1, x_2)^T = (0.99999, 1.00001)^T$, which is only slightly different from its exact value $(1, 1)^T$. We would note similar insensitivity if we were to perturb the other elements of the right-hand-side or elements of the coefficient matrix. We conclude that this problem is well conditioned. On the other hand, the problem

$$\begin{bmatrix} 1.00001 & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 2.00001 \\ 2 \end{bmatrix}$$

is ill conditioned. Its exact solution is $x = (1, 1)^T$, but if we change the first element of the right-hand-side from 2.00001 to 2, the solution would change drastically to $x = (0, 2)^T$.

For general square linear systems $Ax = b$ where $A \in \mathbb{R}^{n\times n}$, the condition number of the matrix (defined in (A.42)) can be used to quantify the conditioning. Specifically, if we perturb A to $\tilde{A}$ and b to $\tilde{b}$ and take $\tilde{x}$ to be the solution of the perturbed system $\tilde{A}\tilde{x} = \tilde{b}$, it can be shown that

$$\frac{\|x - \tilde{x}\|}{\|x\|} \approx \kappa(A) \left[\frac{\|A - \tilde{A}\|}{\|A\|} + \frac{\|b - \tilde{b}\|}{\|b\|} \right]$$

(see, for instance, Golub and Van Loan [115, Section 2.7]). Hence, a large condition number $\kappa(A)$ indicates that the problem $Ax = b$ is ill conditioned, while a modest value indicates well conditioning.

Note that the concept of conditioning has nothing to do with the particular algorithm that is used to solve the problem, only with the numerical problem itself.

Stability, on the other hand, is a property of the algorithm. An algorithm is stable if it is guaranteed to produce accurate answers to all well-conditioned problems in its class, even when floating-point arithmetic is used.

As an example, consider again the linear equations $Ax = b$. We can show that Algorithm A.1, in combination with triangular substitution, yields a computed solution $\tilde{x}$ whose relative error is approximately

$$\frac{\|x - \tilde{x}\|}{\|x\|} \approx \kappa(A) \frac{\text{growth}(A)}{\|A\|} \mathbf{u}, \tag{A.60}$$

where growth(A) is the size of the largest element that arises in A during execution of Algorithm A.1. In the worst case, we can show that growth$(A)/\|A\|$ may be around 2^{n-1}, which indicates that Algorithm A.1 is an unstable algorithm, since even for modest n (say,

$n = 200$), the right-hand-side of (A.60) may be large even when $\kappa(A)$ is modest. In practice, however, large growth factors are rarely observed, so we conclude that Algorithm A.1 is stable for all practical purposes.

Gaussian elimination without pivoting, on the other hand, is definitely unstable. If we omit the possible exchange of rows in Algorithm A.1, the algorithm will fail to produce a factorization even of some well-conditioned matrices, such as

$$A = \begin{bmatrix} 0 & 1 \\ 1 & 2 \end{bmatrix}.$$

For systems $Ax = b$ in which A is symmetric positive definite, the Cholesky factorization in combination with triangular substitution constitutes a stable algorithm for producing a solution x.

References

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*, Prentice-Hall, Englewood Cliffs, N.J., 1993.

[2] H. Akaike, *On a successive transformation of probability distribution and its application to the analysis of the optimum gradient method*, Annals of the Institute of Statistical Mathematics, 11 (1959), pp. 1–17.

[3] M. Al-Baali, *Descent property and global convergence of the Fletcher-Reeves method with inexact line search*, I.M.A. Journal on Numerical Analysis, 5 (1985), pp. 121–124.

[4] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, and D. Sorensen, *LAPACK User's Guide*, SIAM, Philadelphia, 1992.

[5] B. M. Averick, R. G. Carter, J. J. Moré, and G. Xue, *The MINPACK-2 test problem collection*, Preprint MCS–P153–0692, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, Ill., 1992.

[6] P. Baptist and J. Stoer, *On the relation between quadratic termination and convergence properties of minimization algorithms, Part II: Applications*, Numerische Mathematik, 28 (1977), pp. 367–392.

[7] M. Bazaraa, H. Sherali, and C. Shetty, *Nonlinear Programming, Theory and Applications.*, John Wiley & Sons, New York, second ed., 1993.

[8] D. P. BERTSEKAS, *Constrained Optimization and Lagrange Multiplier Methods*, Academic Press, New York, 1982.

[9] ———, *Nonlinear Programming*, Athena Scientific, Belmont, Mass., 1995.

[10] M. BERZ, C. BISCHOF, C. F. CORLISS, AND A. GRIEWANK, eds., *Computational Differentiation: Techniques, Applications, and Tools*, SIAM Publications, Philadelphia, Penn., 1996.

[11] J. R. BIRGE AND F. LOUVEAUX, *Introduction to Stochastic Programming*, Springer-Verlag, New York, 1997.

[12] C. BISCHOF, A. BOUARICHA, P. KHADEMI, AND J. J. MORÉ, *Computing gradients in large-scale optimization using automatic differentiation*, INFORMS Journal on Computing, 9 (1997), pp. 185–194.

[13] C. BISCHOF, A. CARLE, P. KHADEMI, AND A. MAUER, *ADIFOR 2.0: Automatic differentiation of FORTRAN 77 programs*, IEEE Computational Science & Engineering, 3 (1996), pp. 18–32.

[14] C. BISCHOF, G. CORLISS, AND A. GRIEWANK, *Structured second- and higher-order derivatives through univariate Taylor series*, Optimization Methods and Software, 2 (1993), pp. 211–232.

[15] C. BISCHOF AND M. R. HAGHIGHAT, *On hierarchical differentiation*, in Computational Differentiation: Techniques, Applications, and Tools, M. Berz, C. Bischof, G. Corliss, and A. Griewank, eds., SIAM, Philadelphia, 1996, pp. 83–94.

[16] C. BISCHOF, P. KHADEMI, A. BOUARICHA, AND A. CARLE, *Efficient computation of gradients and Jacobians by transparent exploitation of sparsity in automatic differentiation*, Optimization Methods and Software, 7 (1996), pp. 1–39.

[17] C. BISCHOF, L. ROH, AND A. MAUER, *ADIC: An extensible automatic differentiation tool for ANSI-C*, Software—Practice and Experience, 27 (1997), pp. 1427–1456.

[18] Å. BJÖRCK, *Least squares methods*, in Handbook of Numerical Analysis, P. G. Ciarlet and J. L. Lions, eds., Elsevier/North-Holland, Amsterdam, The Netherlands, 1990.

[19] ———, *Numerical Methods for Least Squares Problems*, SIAM Publications, Philadelphia, Penn., 1996.

[20] P. T. BOGGS, R. H. BYRD, AND R. B. SCHNABEL, *A stable and efficient algorithm for nonlinear orthogonal distance regression*, SIAM Journal on Scientific and Statistical Computing, 8 (1987), pp. 1052–1078.

[21] P. T. BOGGS, J. R. DONALDSON, R. H. BYRD, AND R. B. SCHNABEL, *ODRPACK—Software for weighted orthogonal distance regression*, ACM Transactions on Mathematical Software, 15 (1981), pp. 348–364.

[22] P. T. BOGGS AND J. W. TOLLE, *Convergence properties of a class of rank-two updates*, SIAM Journal on Optimization, 4 (1994), pp. 262–287.

[23] ———, *Sequential quadratic programming*, Acta Numerica, 4 (1996), pp. 1–51.

[24] P. T. BOGGS, J. W. TOLLE, AND P. WANG, *On the local convergence of quasi-Newton methods for constrained optimization*, SIAM Journal on Control and Optimization, 20 (1982), pp. 161–171.

[25] I. BONGARTZ, A. R. CONN, N. I. M. GOULD, AND P. L. TOINT, *CUTE: Constrained and unconstrained testing environment*, Research Report, IBM T.J. Watson Research Center, Yorktown Heights, NY, 1993.

[26] S. BOYD, L. EL GHAOUI, E. FERON, AND V. BALAKRISHNAN, *Linear Matrix Inequalities in Systems and Control Theory*, SIAM Publications, Phildelphia, 1994.

[27] R. P. BRENT, *Algorithms for minimization without derivatives*, Prentice Hall, Englewood Cliffs, N.J., 1973.

[28] A. BUCKLEY AND A. LENIR, *QN-like variable storage conjugate gradients*, Mathematical Programming, 27 (1983), pp. 155–175.

[29] R. BULIRSCH AND J. STOER, *Introduction to Numerical Analysis*, Springer-Verlag, New York, 1980.

[30] J. R. BUNCH AND L. KAUFMAN, *Some stable methods for calculating inertia and solving symmetric linear systems*, Mathematics of Computation, 31 (1977), pp. 163–179.

[31] J. R. BUNCH AND B. N. PARLETT, *Direct methods for solving symmetric indefinite systems of linear equations*, SIAM Journal on Numerical Analysis, 8 (1971), pp. 639–655.

[32] J. V. BURKE AND J. J. MORÉ, *Exposing constraints*, SIAM Journal on Optimization, 4 (1994), pp. 573–595.

[33] W. BURMEISTER, *Die Konvergenzordnung des Fletcher-Powell Algorithmus*, Z. Angew. Math. Mech., 53 (1973), pp. 693–699.

[34] R. H. BYRD, M. E. HRIBAR, AND J. NOCEDAL, *An interior-point algorithm for large-scale nonlinear programming*, Technical Report 97/05, Optimization Technology Center, Argonne National Laboratory and Northwestern University, July 1997.

[35] R. H. BYRD, H. F. KHALFAN, AND R. B. SCHNABEL, *Analysis of a symmetric rank-one trust region method*, SIAM Journal on Optimization, 6 (1996), pp. 1025–1039.

[36] R. H. BYRD AND J. NOCEDAL, *An analysis of reduced Hessian methods for constrained optimization*, Mathematical Programming, 49 (1991), pp. 285–323.

[37] R. H. BYRD, J. NOCEDAL, AND R. B. SCHNABEL, *Representations of quasi-Newton matrices and their use in limited-memory methods*, Mathematical Programming, Series A, 63 (1994), pp. 129–156.

[38] R. H. BYRD, J. NOCEDAL, AND Y. YUAN, *Global convergence of a class of quasi-Newton methods on convex problems*, SIAM Journal on Numerical Analysis, 24 (1987), pp. 1171–1190.

[39] R. H. BYRD, R. B. SCHNABEL, AND G. A. SCHULTZ, *Approximate solution of the trust regions problem by minimization over two-dimensional subspaces*, Mathematical Programming, 40 (1988), pp. 247–263.

[40] R. CHAMBERLAIN, C. LEMARÉCHAL, H. C. PEDERSEN, AND M. J. D. POWELL, *The watchdog technique for forcing convergence in algorithms for constrained optimization*, Mathematical Programming, 16 (1982), pp. 1–17.

[41] S. H. CHENG AND H. J. HIGHAM, *A modified Cholesky algorithm based on a symmetric indefinite factorization*, technical report, University of Manchester, 1996.

[42] F. H. CLARKE, *Optimization and Nonsmooth Analysis*, John Wiley & Sons, New York, 1983.

[43] A. COHEN, *Rate of convergence of several conjugate gradient algorithms*, SIAM Journal on Numerical Analysis, 9 (1972), pp. 248–259.

[44] T. F. COLEMAN AND A. R. CONN, *Non-linear programming via an exact penalty-function: Asymptotic analysis*, Mathematical Programming, 24 (1982), pp. 123–136.

[45] T. F. COLEMAN AND A. R. CONN, *On the local convergence of a quasi-Newton method for the nonlinear programming problem*, SIAM Journal on Numerical Analysis, 21 (1984), pp. 755–769.

[46] T. F. COLEMAN, B. GARBOW, AND J. J. MORÉ, *Software for estimating sparse Jacobian matrices*, ACM Transactions on Mathematical Software, 10 (1984), pp. 329–345.

[47] ———, *Software for estimating sparse Hessian matrices*, ACM Transactions on Mathematical Software, 11 (1985), pp. 363–377.

[48] T. F. COLEMAN AND J. J. MORÉ, *Estimation of sparse Jacobian matrices and graph coloring problems*, SIAM Journal on Numerical Analysis, 20 (1983), pp. 187–209.

[49] ———, *Estimation of sparse Hessian matrices and graph coloring problems*, Mathematical Programming, 28 (1984), pp. 243–270.

[50] T. F. COLEMAN AND D. C. SORENSEN, *A note on the computation of an orthonormal basis for the null space of a matrix*, Mathematical Programming, 29 (1984), pp. 234–242.

[51] A. R. CONN, N. I. M. GOULD, AND P. L. TOINT, *Testing a class of algorithms for solving minimzation problems with simple bounds on the variables*, Mathematics of Computation, 50 (1988), pp. 399–430.

[52] ———, *Convergence of quasi-Newton matrices generated by the symmetric rank one update*, Mathematical Programming, 50 (1991), pp. 177–195.

[53] ———, *LANCELOT: a FORTRAN package for large-scale nonlinear optimization (Release A)*, no. 17 in Springer Series in Computational Mathematics, Springer-Verlag, New York, 1992.

[54] ———, *Numerical experiments with the LANCELOT package (Release A) for large-scale nonlinear optimization*, Report 92/16, Department of Mathematics, University of Namur, Belgium, 1992.

[55] ———, *A note on using alternative second-order models for the subproblems arising in barrier function methods for minimization*, Numerische Mathematik, 68 (1994), pp. 17–33.

[56] W. J. COOK, W. H. CUNNINGHAM, W. R. PULLEYBLANK, AND A. SCHRIJVER, *Combinatorial Optimization*, John Wiley & Sons, New York, 1997.

[57] B. F. CORLISS AND L. B. RALL, *An introduction to automatic differentiation*, in Computational Differentiation: Techniques, Applications, and Tools, M. Berz, C. Bischof, G. F. Corliss, and A. Griewank, eds., SIAM Publications, Philadelphia, Penn., 1996, ch. 1.

[58] T. H. CORMEN, C. E. LEISSERSON, AND R. L. RIVEST, *Introduction to Algorithms*, MIT Press, 1990.

[59] R. W. COTTLE, J.-S. PANG, AND R. E. STONE, *The Linear Complementarity Problem*, Academic Press, San Diego, 1992.

[60] R. COURANT, *Variational methods for the solution of problems with equilibrium and vibration*, Bull. Amer. Math. Soc., 49 (1943), pp. 1–23.

[61] H. P. CROWDER AND P. WOLFE, *Linear convergence of the conjugate gradient method*, IBM Journal of Research and Development, 16 (1972), pp. 431–433.

[62] A. CURTIS, M. J. D. POWELL, AND J. REID, *On the estimation of sparse Jacobian matrices*, Journal of the Institute of Mathematics and its Applications, 13 (1974), pp. 117–120.

[63] G. B. DANTZIG, *Linear Programming and Extensions*, Princeton University Press, Princeton, New Jersey, 1963.

[64] W. C. DAVIDON, *Variable metric method for minimization*, Technical Report ANL–5990 (revised), Argonne National Laboratory, Argonne, Il, 1959.

[65] ———, *Variable metric method for minimization*, SIAM Journal on Optimization, 1 (1991), pp. 1–17.

[66] R. S. DEMBO, S. C. EISENSTAT, AND T. STEIHAUG, *Inexact Newton methods*, SIAM Journal on Numerical Analysis, 19 (1982), pp. 400–408.

[67] J. E. DENNIS, D. M. GAY, AND R. E. WELSCH, *Algorithm 573 — NL2SOL, An adaptive nonlinear least-squares algorithm*, ACM Transactions on Mathematical Software, 7 (1981), pp. 348–368.

[68] J. E. DENNIS AND J. J. MORÉ, *Quasi-Newton methods, motivation and theory*, SIAM Review, 19 (1977), pp. 46–89.

[69] J. E. DENNIS AND R. B. SCHNABEL, *Numerical Methods for Unconstrained Optimization*, Prentice-Hall, Englewood Cliffs, NJ, 1983. Reprinted by SIAM Publications, 1993.

[70] J. E. DENNIS AND R. B. SCHNABEL, *A view of unconstrained optimization*, in Optimization, vol. 1 of Handbooks in Operations Research and Management, Elsevier Science Publishers, Amsterdam, the Netherlands, 1989, pp. 1–72.

[71] P. DEUFLHARD, R. W. FREUND, AND A. WALTER, *Fast secant methods for the iterative solution of large nonsymmetric linear systems*, Impact of Computing in Science and Engineering, 2 (1990), pp. 244–276.

[72] I. I. DIKIN, *Iterative solution of problems of linear and quadratic programming*, Soviet Mathematics-Doklady, 8 (1967), pp. 674–675.

[73] I. S. DUFF, J. NOCEDAL, AND J. K. REID, *The use of linear programming for the solution of sparse sets of nonlinear equations*, SIAM Journal on Scientific and Statistical Computing, 8 (1987), pp. 99–108.

[74] I. S. DUFF AND J. K. REID, *The multifrontal solution of indefinite sparse symmetric linear equations*, ACM Transactions on Mathematical Software, 9 (1983), pp. 302–325.

[75] ———, *The design of MA48, a code for direct solution of sparse unsymmetric linear systems of equations*, ACM Transactions on Mathematical Software, 22 (1996), pp. 187–226.

[76] I. S. DUFF, J. K. REID, N. MUNKSGAARD, AND H. B. NEILSEN, *Direct solution of sets of linear equations whose matrix is sparse symmetric and indefinite*, Journal of the Institute of Mathematics and its Applications, 23 (1979), pp. 235–250.

[77] J. C. DUNN, *A projected Newton method for minimization problems with nonlinear inequality constraints*, Numerische Mathematik, 53 (1988), pp. 377–409.

[78] J. DUSSAULT, *Numerical stability and efficiency of penalty algorithms*, SIAM Journal on Numerical Analysis, 32 (1995), pp. 296–317.

[79] A. V. FIACCO AND G. P. MCCORMICK, *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*, John Wiley & Sons, New York, NY, 1968. reprinted by SIAM Publications, 1990.

[80] R. FLETCHER, *A general quadratic programming algorithm*, Journal of the Institute of Mathematics and its Applications, 7 (1971), pp. 76–91.

[81] ———, *A class of methods for nonlinear programming, iii: Rates of convergence*, in Numerical Methods for Non-Linear Optimization, F. A. Lootsma, ed., Academic Press, London and New York, 1972, pp. 371–382.

[82] ———, *Second order corrections for non-differentiable optimization*, in Numerical Analysis, D. Griffiths, ed., Springer Verlag, 1982, pp. 85–114. Proceedings Dundee 1981.

[83] ———, *Practical Methods of Optimization*, John Wiley & Sons, New York, second ed., 1987.

[84] ———, *An optimal positive definite update for sparse Hessian matrices*, SIAM Journal on Optimization, 5 (1995), pp. 192–218.

[85] R. FLETCHER, A. GROTHEY, AND S. LEYFFER, *Computing sparse Hessian and Jacobian approximations with optimal hereditary properties*, technical report, Department of Mathematics, University of Dundee, 1996.

[86] R. FLETCHER AND S. LEYFFER, *Nonlinear programming without a penalty function*, Tech. Rep. NA/171, Department of Mathematics, University of Dundee, September 1997.

[87] R. FLETCHER AND A. P. MCCANN, *Acceleration techniques for nonlinear programming*, in Optimization, R. Fletcher, ed., Academic Press, London, 1969, pp. 203–214.

[88] R. FLETCHER AND C. M. REEVES, *Function minimization by conjugate gradients*, Computer Journal, 7 (1964), pp. 149–154.

[89] R. FLETCHER AND E. SAINZ DE LA MAZA, *Nonlinear programming and nonsmooth optimization by successive linear programming*, Mathematical Programming, (1989), pp. 235–256.

[90] C. FLOUDAS AND P. PARDALOS, eds., *Recent Advances in Global Optimization*, Princeton University Press, Princeton, NJ, 1992.

[91] A. FORSGREN AND P. E. GILL, *Primal-dual interior methods for nonconvex nonlinear programming*, SIAM Journal on Optimization, 8 (1998), pp. 1132–1152.

[92] R. FOURER, D. M. GAY, AND B. W. KERNIGHAN, *AMPL: A Modeling Language for Mathematical Programming*, The Scientific Press, South San Francisco, Calif., 1993.

[93] R. FOURER AND S. MEHROTRA, *Solving symmetric indefinite systems in an interior-point method for linear programming*, Mathematical Programming, 62 (1993), pp. 15–39.

[94] R. FREUND AND N. NACHTIGAL, *QMR: A quasi-minimal residual method for non-Hermitian linear systems*, Numerische Mathematik, 60 (1991), pp. 315–339.

[95] K. R. FRISCH, *The logarithmic potential method of convex programming*, Technical Report, University Institute of Economics, Oslo, Norway, 1955.

[96] D. GABAY, *Reduced quasi-Newton methods with feasibility improvement for nonlinearly constrained optimization*, Mathematical Programming Studies, 16 (1982), pp. 18–44.

[97] U. M. GARCIA-PALOMARES AND O. L. MANGASARIAN, *Superlinearly convergent quasi-Newton methods for nonlinearly constrained optimization problems*, Mathematical Programming, 11 (1976), pp. 1–13.

[98] D. M. GAY, *More AD of nonlinear AMPL models: computing Hessian information and exploiting partial separability*. To appear in the Proceedings of the Second International Workshop on Computational Differentiation, 1996.

[99] D. M. GAY, M. L. OVERTON, AND M. H. WRIGHT, *A primal-dual interior method for nonconvex nonlinear programming*, Technical Report 97-4-08, Computing Sciences Research Center, Bell Laboratories, Murray Hill, N.J., July 1997.

[100] R.-P. GE AND M. J. D. POWELL, *The convergence of variable metric matrices in unconstrained optimization*, Mathematical Programming, 27 (1983), pp. 123–143.

[101] J. GILBERT, *Maintaining the positive definiteness of the matrices in reduced secant methods for equality constrained optimization*, Mathematical Programming, 50 (1991), pp. 1–28.

[102] J. GILBERT AND C. LEMARÉCHAL, *Some numerical experiments with variable-storage quasi-Newton algorithms*, Mathematical Programming, Series B, 45 (1989), pp. 407–435.

[103] J. GILBERT AND J. NOCEDAL, *Global convergence properties of conjugate gradient methods for optimization*, SIAM Journal on Optimization, 2 (1992), pp. 21–42.

[104] P. GILL, W. MURRAY, AND M. H. WRIGHT, *Practical Optimization*, Academic Press, 1981.

[105] P. E. GILL, G. H. GOLUB, W. MURRAY, AND M. A. SAUNDERS, *Methods for modifying matrix factorizations*, Mathematics of Computation, 28 (1974), pp. 505–535.

[106] P. E. GILL AND M. W. LEONARD, *Limited-memory reduced-Hessian methods for unconstrained optimization*, Numerical Analysis Report NA 97-1, University of California, San Diego, 1997.

[107] P. E. GILL AND W. MURRAY, *Numerically stable methods for quadratic programming*, Mathematical Programming, 14 (1978), pp. 349–372.

[108] P. E. GILL, W. MURRAY, AND M. A. SAUNDERS, *User's guide for SNOPT (Version 5.3): A FORTRAN package for large-scale nonlinear programming*, Technical Report NA 97-4, Department of Mathematics, University of California, San Diego, 1997.

[109] P. E. GILL, W. MURRAY, M. A. SAUNDERS, G. W. STEWART, AND M. H. WRIGHT, *Properties of a representation of a basis for the null space*, Mathematical Programming, 33 (1985), pp. 172–186.

[110] P. E. GILL, W. MURRAY, M. A. SAUNDERS, AND M. H. WRIGHT, *User's guide for SOL/QPSOL*, Technical Report SOL84–6, Department of Operations Research, Stanford University, Stanford, California, 1984.

[111] ———, *User's guide for NPSOL (Version 4.0): A FORTRAN package for nonlinear programming*, Technical Report SOL 86-2, Department of Operations Research, Stanford University, Stanford, CA, 1986.

[112] P. E. GILL, W. MURRAY, M. A. SAUNDERS, AND M. H. WRIGHT, *Constrained nonlinear programming*, in Optimization, vol. 1 of Handbooks in Operations Research and Management, Elsevier Science Publishers, Amsterdam, the Netherlands, 1989, pp. 171–210.

[113] D. GOLDFARB, *Curvilinear path steplength algorithms for minimization which use directions of negative curvature*, Mathematical Programming, 18 (1980), pp. 31–40.

[114] D. GOLDFARB AND J. FORREST, *Steepest edge simplex algorithms for linear programming*, Mathematical Programming, 57 (1992), pp. 341–374.

[115] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, The Johns Hopkins University Press, Baltimore, 3rd ed., 1996.

[116] J. GONDZIO, *Multiple centrality corrections in a primal-dual method for linear programming*, Computational Optimization and Applications, 6 (1996), pp. 137–156.

[117] J. GOODMAN, *Newton's method for constrained optimization*, Mathematical Programming, 33 (1985), pp. 162–171.

[118] N. I. M. GOULD, *On the accurate determination of search directions for simple differentiable penalty functions*, I.M.A. Journal on Numerical Analysis, 6 (1986), pp. 357–372.

[119] ———, *On the convergence of a sequential penalty function method for constrained minimization*, SIAM Journal on Numerical Analysis, 26 (1989), pp. 107–128.

[120] ———, *An algorithm for large scale quadratic programming*, I.M.A. Journal on Numerical Analysis, 11 (1991), pp. 299–324.

[121] N. I. M. GOULD, S. LUCIDI, M. ROMA, AND P. L. TOINT, *Solving the trust-region subproblem using the Lanczos method.* To appear in SIAM Journal on Optimization, 1998.

[122] A. GRIEWANK, *Achieving logarithmic growth of temporal and spatial complexity in reverse automatic differentiation*, Optimization Methods and Software, 1 (1992), pp. 35–54.

[123] ———, *Automatic directional differentiation of nonsmooth composite functions*, in Seventh French-German Conference on Optimization, 1994.

[124] ———, *Computational Differentiation and Optimization*, in Mathematical Programming: State of the Art 1994, J. R. Birge and K. G. Murty, eds., The University of Michigan, Michigan, USA, 1994, pp. 102–131.

[125] A. GRIEWANK AND G. F. CORLISS, eds., *Automatic Differentition of Algorithms*, SIAM Publications, Philadelphia, Penn., 1991.

[126] A. GRIEWANK, D. JUEDES, AND J. UTKE, *ADOL-C, A package for the automatic differentiation of algorithms written in C/C++*, ACM Transactions on Mathematical Software, 22 (1996), pp. 131–167.

[127] A. GRIEWANK AND P. L. TOINT, *Local convergence analysis of partitioned quasi-Newton updates*, Numerische Mathematik, 39 (1982), pp. 429–448.

[128] ———, *On the unconstrained optimization of partially separable objective functions*, in Nonlinear Optimization 1981, M. J. D. Powell, ed., Academic Press, London, 1982, pp. 301–312.

[129] ———, *Partitioned variable metric updates for large structured optimization problems*, Numerische Mathematik, 39 (1982), pp. 119–137.

[130] J. GRIMM, L. POTTIER, AND N. ROSTAING-SCHMIDT, *Optimal time and minimum space time product for reversing a certain class of programs*, in Computational Differentiation, Techniques, Applications, and Tools, M. Berz, C. Bischof, G. Corliss, and A. Griewank, eds., SIAM, Philadelphia, 1996, pp. 95–106.

[131] S. P. HAN, *Superlinearly convergent variable metric algorithms for general nonlinear programming problems*, Mathematical Programming, 11 (1976), pp. 263–282.

[132] ———, *A globally convergent method for nonlinear programming*, Journal of Optimization Theory and Applications, 22 (1977), pp. 297–309.

[133] *Harwell Subroutine Library, Release 10*, Advanced Computing Department, AEA Industrial Technology, Harwell Laboratory, Oxfordshire, United Kingdom, 1990.

[134] M. R. HESTENES, *Multiplier and gradient methods*, Journal of Optimization Theory and Applications, 4 (1969), pp. 303–320.

[135] M. R. HESTENES AND E. STIEFEL, *Methods of conjugate gradients for solving linear systems*, Journal of Research of the National Bureau of Standards, 49 (1952), pp. 409–436.

[136] N. J. HIGHAM, *Accuracy and Stability of Numerical Algorithms*, SIAM Publications, Philadelphia, 1996.

[137] J.-B. HIRIART-URRUTY AND C. LEMARÉCHAL, *Convex Analysis and Minimization Algorithms*, Springer-Verlag, Berlin, New York, 1993.

[138] J. E. DENNIS, JR. AND R. B. SCHNABEL, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, Englewood Cliffs, N.J., 1983.

[139] P. KALL AND S. W. WALLACE, *Stochastic Programming*, John Wiley & Sons, New York, 1994.

[140] N. KARMARKAR, *A new polynomial-time algorithm for linear programming*, Combinatorics, 4 (1984), pp. 373–395.

[141] C. T. KELLEY, *Iterative Methods for Linear and Nonlinear Equations*, SIAM Publications, Philadelphia, Penn., 1995.

[142] L. G. KHACHIYAN, *A polynomial algorithm in linear programming*, Soviet Mathematics Doklady, 20 (1979), pp. 191–194.

[143] H. F. KHALFAN, R. H. BYRD, AND R. B. SCHNABEL, *A theoretical and experimental study of the symmetric rank one update*, SIAM Journal on Optimization, 3 (1993), pp. 1–24.

[144] V. KLEE AND G. J. MINTY, *How good is the simplex algorithm?* in Inequalities, O. Shisha, ed., Academic Press, New York, 1972, pp. 159–175.

[145] H. W. KUHN AND A. W. TUCKER, *Nonlinear programming*, in Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability, J. Neyman, ed., Berkeley, CA, 1951, University of California Press, pp. 481–492.

[146] M. LALEE, J. NOCEDAL, AND T. PLANTENGA, *On the implementation of an algorithm for large-scale equality constrained optimization*, SIAM Journal on Optimization, (1998), pp. 682–706.

[147] S. LANG, *Real Analysis*, Addison-Wesley, Reading, MA, second ed., 1983.

[148] C. L. LAWSON AND R. J. HANSON, *Solving Least Squares Problems*, Prentice-Hall, Englewood Cliffs, NJ, 1974.

[149] C. LEMARÉCHAL, *A view of line searches*, in Optimization and Optimal Control, W. Oettli and J. Stoer, eds., no. 30 in Lecture Notes in Control and Information Science, Springer-Verlag, 1981, pp. 59–78.

[150] K. LEVENBERG, *A method for the solution of certain non-linear problems in least squares*, Quarterly of Applied Mathematics, 2 (1944), pp. 164–168.

[151] D. C. LIU AND J. NOCEDAL, *On the limited-memory BFGS method for large scale optimization*, Mathematical Programming, 45 (1989), pp. 503–528.

[152] D. LUENBERGER, *Introduction to Linear and Nonlinear Programming*, Addison Wesley, second ed., 1984.

[153] *Macsyma User's Guide*, second ed., 1996.

[154] O. L. MANGASARIAN, *Nonlinear Programming*, McGraw-Hill, New York, 1969. Reprinted by SIAM Publications, 1995.

[155] O. L. MANGASARIAN AND L. L. SCHUMAKER, *Discrete splines via mathematical programming*, SIAM Journal on Control, 9 (1971), pp. 174–183.

[156] N. MARATOS, *Exact penalty function algorithms for finite dimensional and control optimization problems*, Ph.D. thesis, University of London, 1978.

[157] H. M. MARKOWITZ, *Portfolio selection*, Journal of Finance, 8 (1952), pp. 77–91.

[158] ———, *The elimination form of the inverse and its application to linear programming*, Management Science, 3 (1957), pp. 255–269.

[159] ———, *Portfolio Selection: Efficient Diversification of Investments*, Basil Blackwell, Cambridge, Mass., 1991.

[160] D. W. MARQUARDT, *An algorithm for least squares estimation of non-linear parameters*, SIAM Journal, 11 (1963), pp. 431–441.

[161] D. Q. MAYNE AND E. POLAK, *A superlinearly convergent algorithm for constrained optimization problems*, Mathematical Programming Studies, 16 (1982), pp. 45–61.

[162] L. MCLINDEN, *An analogue of Moreau's proximation theorem, with applications to the nonlinear complementarity problem*, Pacific Journal of Mathematics, 88 (1980), pp. 101–161.

[163] N. MEGIDDO, *Pathways to the optimal set in linear programming*, in Progress in Mathematical Programming: Interior-Point and Related Methods, N. Megiddo, ed., Springer-Verlag, New York, N.Y., 1989, ch. 8, pp. 131–158.

[164] S. MEHROTRA, *On the implementation of a primal-dual interior point method*, SIAM Journal on Optimization, 2 (1992), pp. 575–601.

[165] S. MIZUNO, M. TODD, AND Y. YE, *On adaptive step primal-dual interior-point algorithms for linear programming*, Mathematics of Operations Research, 18 (1993), pp. 964–981.

[166] J. J. MORÉ, *The Levenberg-Marquardt algorithm: Implementation and theory*, in Lecture Notes in Mathematics, No. 630–Numerical Analysis, G. Watson, ed., Springer-Verlag, 1978, pp. 105–116.

[167] ———, *Recent developments in algorithms and software for trust region methods*, in Mathematical Programming: The State of the Art, Springer-Verlag, Berlin, 1983, pp. 258–287.

[168] ———, *A collection of nonlinear model problems*, in Computational Solution of Nonlinear Systems of Equations, vol. 26 of Lectures in Applied Mathematics, American Mathematical Society, Providence, R.I., 1990, pp. 723–762.

[169] J. J. MORÉ AND D. C. SORENSEN, *On the use of directions of negative curvature in a modified Newton method*, Mathematical Programming, 16 (1979), pp. 1–20.

[170] ———, *Computing a trust region step*, SIAM Journal on Scientific and Statistical Computing, 4 (1983), pp. 553–572.

[171] ———, *Newton's method*, in Studies in Numerical Analysis, vol. 24 of MAA Studies in Mathematics, The Mathematical Association of America, 1984, pp. 29–82.

[172] J. J. MORÉ AND D. J. THUENTE, *Line search algorithms with guaranteed sufficient decrease*, ACM Transactions on Mathematical Software, 20 (1994), pp. 286–307.

[173] J. J. MORÉ AND S. J. WRIGHT, *Optimization Software Guide*, SIAM Publications, Philadelphia, 1993.

[174] W. MURRAY AND M. H. WRIGHT, *Line search procedures for the logarithmic barrier function*, SIAM Journal on Optimization, 4 (1994), pp. 229–246.

[175] B. A. MURTAGH AND M. A. SAUNDERS, *MINOS 5.1 User's guide*, Technical Report SOL-83-20R, Stanford University, 1987.

[176] K. G. MURTY AND S. N. KABADI, *Some NP-complete problems in quadratic and nonlinear programming*, Mathematical Programming, 19 (1987), pp. 200–212.

[177] S. G. NASH, *Newton-type minimization via the Lanczos method*, SIAM Journal on Numerical Analysis, 21 (1984), pp. 553–572.

[178] ———, *SUMT (Revisited)*, Operations Research, 46 (1998), pp. 763–775.

[179] G. L. NEMHAUSER AND L. A. WOLSEY, *Integer and Combinatorial Optimization*, John Wiley & Sons, New York, 1988.

[180] A. S. NEMIROVSKII AND D. B. YUDIN, *Problem complexity and method efficiency*, John Wiley & Sons, New York, 1983.

[181] Y. E. NESTEROV AND A. S. NEMIROVSKII, *Interior Point Polynomial Methods in Convex Programming*, SIAM Publications, Philadelphia, 1994.

[182] G. N. NEWSAM AND J. D. RAMSDELL, *Estimation of sparse Jacobian matrices*, SIAM Journal on Algebraic and Discrete Methods, 4 (1983), pp. 404–418.

[183] J. NOCEDAL, *Updating quasi-Newton matrices with limited storage*, Mathematics of Computation, 35 (1980), pp. 773–782.

[184] ———, *Theory of algorithms for unconstrained optimization*, Acta Numerica, 1 (1992), pp. 199–242.

[185] J. M. ORTEGA AND W. C. RHEINBOLDT, *Iterative solution of nonlinear equations in several variables*, Academic Press, New York and London, 1970.

[186] M. R. OSBORNE, *Nonlinear least squares—the Levenberg algorithm revisited*, Journal of the Australian Mathematical Society, Series B, 19 (1976), pp. 343–357.

[187] ———, *Finite Algorithms in Optimization and Data Analysis*, John Wiley & Sons, 1985.

[188] C. C. PAIGE AND M. A. SAUNDERS, *LSQR: An algorithm for sparse linear equations and sparse least squares*, ACM Transactions on Mathematical Software, 8 (1982), pp. 43–71.

[189] E. R. PANIER AND A. L. TITS, *On combining feasibility, descent and superlinear convergence in inequality constrained optimization*, Mathematical Programming, 59 (1993), pp. 261–276.

[190] C. H. PAPADIMITRIOU AND K. STEIGLITZ, *Combinatorial Optimization: Algorithms and Complexity*, Prentice Hall, Englewood Cliffs, NJ, 1982.

[191] R. J. PLEMMONS, *Least squares computations for geodetic and related problems*, in High Speed Computing, R. Williamson, ed., University of Illinois Press, 1989, pp. 198–200.

[192] R. J. PLEMMONS AND R. WHITE, *Substructuring methods for computing the nullspace of equilibrium matrices*, SIAM Journal on Matrix Analysis and Applications, 11 (1990), pp. 1–22.

[193] E. POLAK, *Optimization: Algorithms and Consistent Approximations*, no. 124 in Applied Mathematical Sciences, Springer, 1997.

[194] E. POLAK AND G. RIBIÈRE, *Note sur la convergence de méthodes de directions conjuguées*, Revue Française d'Informatique et de Recherche Opérationnelle, 16 (1969), pp. 35–43.

[195] M. J. D. POWELL, *A method for nonlinear constraints in minimization problems*, in Optimization, R. Fletcher, ed., Academic Press, New York, NY, 1969, pp. 283–298.

[196] ———, *A hybrid method for nonlinear equations*, in Numerical Methods for Nonlinear Algebraic Equations, P. Rabinowitz, ed., Gordon & Breach, London, 1970, pp. 87–114.

[197] ———, *Problems related to unconstrained optimization*, in Numerical Methods for Unconstrained Optimization, W. Murray, ed., Academic Press, 1972, pp. 29–55.

[198] ———, *On search directions for minimization algorithms*, Mathematical Programming, 4 (1973), pp. 193–201.

[199] ———, *Convergence properties of a class of minimization algorithms*, in Nonlinear Programming 2, O. L. Mangasarian, R. R. Meyer, and S. M. Robinson, eds., Academic Press, New York, 1975, pp. 1–27.

[200] ———, *Some convergence properties of the conjugate gradient method*, Mathematical Programming, 11 (1976), pp. 42–49.

[201] ———, *Some global convergence properties of a variable metric algorithm for minimization without exact line searches*, in Nonlinear Programming, SIAM-AMS Proceedings, Vol. IX, R. W. Cottle and C. E. Lemke, eds., SIAM Publications, 1976, pp. 53–72.

[202] ———, *A fast algorithm for nonlinearly constrained optimization calculations*, in Numerical Analysis Dundee 1977, G. A. Watson, ed., Springer Verlag, Berlin, 1977, pp. 144–157.

[203] ———, *Restart procedures for the conjugate gradient method*, Mathematical Programming, 12 (1977), pp. 241–254.

[204] ———, *Algorithms for nonlinear constraints that use Lagrangian functions*, Mathematical Programming, 14 (1978), pp. 224–248.

[205] ———, *The convergence of variable metric methods for nonlinearly constrained optimization calculations*, in Nonlinear Programming 3, Academic Press, New York and London, 1978, pp. 27–63.

[206] ———, *On the rate of convergence of variable metric algorithms for unconstrained optimization*, Technical Report DAMTP 1983/NA7, Department of Applied Mathematics and Theoretical Physics, Cambridge University, 1983.

[207] ———, *Nonconvex minimization calculations and the conjugate gradient method*, Lecture Notes in Mathematics, 1066 (1984), pp. 122–141.

[208] ———, *The performance of two subroutines for constrained optimization on some difficult test problems*, in Numerical Optimization, P. T. Boggs, R. H. Byrd, and R. B. Schnabel, eds., SIAM Publications, Philadelphia, 1984.

[209] ———, *Convergence properties of algorithms for nonlinear optimization*, SIAM Review, 28 (1986), pp. 487–500.

[210] M. J. D. POWELL AND P. L. TOINT, *On the estimation of sparse Hessian matrices*, SIAM Journal on Numerical Analysis, 16 (1979), pp. 1060–1074.

[211] D. RALPH AND S. J. WRIGHT, *Superlinear convergence of an interior-point method for monotone variational inequalities*, in Complementarity and Variational Problems: State of the Art, SIAM Publications, Philadelphia, Penn., 1997, pp. 345–385.

[212] Z. REN AND K. MOFFATT, *Quantitative analysis of synchotron Laue diffraction patterns in macromolecular crystallography*, Journal of Applied Crystallography, 28 (1995), pp. 461–481.

[213] J. M. RESTREPO, G. K. LEAF, AND A. GRIEWANK, *Circumventing storage limitations in variational data assimilation studies*, Preprint ANL/MCS-P515-0595, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, Ill., 1995.

[214] K. RITTER, *On the rate of superlinear convergence of a class of variable metric methods*, Numerische Mathematik, 35 (1980), pp. 293–313.

[215] S. M. ROBINSON, *A quadratically convergent algorithm for general nonlinear programming problems*, Mathematical Programming, (1972), pp. 145–156.

[216] R. T. ROCKAFELLAR, *The multiplier method of Hestenes and Powell applied to convex programming*, Journal of Optimization Theory and Applications, 12 (1973), pp. 555–562.

[217] ———, *Lagrange multipliers and optimality*, SIAM Review, 35 (1993), pp. 183–238.

[218] J. B. ROSEN AND J. KREUSER, *A gradient projection algorithm for nonlinear constraints*, in Numerical Methods for Non-Linear Optimization, F. A. Lootsma, ed., Academic Press, London and New York, 1972, pp. 297–300.

[219] N. ROSTAING, S. DALMAS, AND A. GALLIGO, *Automatic differentiation in Odyssee*, Tellus, 45a (1993), pp. 558–568.

[220] Y. SAAD, *Iterative Methods for Sparse Linear Systems*, PWS Publishing Company, 1996.

[221] Y. SAAD AND M. SCHULTZ, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM Journal on Scientific and Statistical Computing, 7 (1986), pp. 856–869.

[222] K. SCHITTKOWSKI, *The nonlinear programming method of Wilson, Han and Powell with an augmented Lagrangian type line search function*, Numerische Mathematik, (1981), pp. 83–114.

[223] R. B. SCHNABEL AND E. ESKOW, *A new modified Cholesky factorization*, SIAM Journal on Scientific Computing, 11 (1991), pp. 1136–1158.

[224] R. B. SCHNABEL AND P. D. FRANK, *Tensor methods for nonlinear equations*, SIAM Journal on Numerical Analysis, 21 (1984), pp. 815–843.

[225] G. SCHULLER, *On the order of convergence of certain quasi-Newton methods*, Numerische Mathematik, 23 (1974), pp. 181–192.

[226] G. A. SCHULTZ, R. B. SCHNABEL, AND R. H. BYRD, *A family of trust-region-based algorithms for unconstrained minimization with strong global convergence properties*, SIAM Journal on Numerical Analysis, 22 (1985), pp. 47–67.

[227] G. A. F. SEBER AND C. J. WILD, *Nonlinear Regression*, John Wiley & Sons, New York, 1989.

[228] D. F. SHANNO AND K. H. PHUA, *Remark on Algorithm 500: Minimization of unconstrained multivariate functions*, ACM Transactions on Mathematical Software, 6 (1980), pp. 618–622.

[229] A. SHERMAN, *On Newton-iterative methods for the solution of systems of nonlinear equations*, SIAM Journal on Numerical Analysis, 15 (1978), pp. 755–771.

[230] D. D. SIEGEL, *Implementing and modifying Broyden class updates for large scale optimization*,, Technical Report AMTP 1992/NA12, Department of Applied Mathematics and Theoretical Physics, University of Cambridge, 1992.

[231] T. STEIHAUG, *The conjugate gradient method and trust regions in large scale optimization*, SIAM Journal on Numerical Analysis, 20 (1983), pp. 626–637.

[232] J. STOER, *On the relation between quadratic termination and convergence properties of minimization algorithms. Part I: Theory*, Numerische Mathematik, 28 (1977), pp. 343–366.

[233] K. TANABE, *Centered Newton method for mathematical programming*, in System Modeling and Optimization: Proceedings of the 13th IFIP conference, vol. 113 of Lecture Notes in Control and Information Systems, Berlin, 1988, Springer-Verlag, pp. 197–206.

[234] R. A. TAPIA, *Quasi-Newton methods for equality constrained optimization: Equivalence of existing methods and a new implementation*, in Nonlinear Programming 3 (O. Mangasarian, R. Meyer, and S. Robinson, eds), Academic Press, New York, NY, (1978) pp. 125–164.

[235] M. J. TODD, *Potential reduction methods in mathematical programming*, Mathematical Programming, Series B, 76 (1997), pp. 3–45.

[236] M. J. TODD AND Y. YE, *A centered projective algorithm for linear programming*, Mathematics of Operations Research, 15 (1990), pp. 508–529.

[237] P. L. TOINT, *On sparse and symmetric matrix updating subject to a linear equation*, Mathematics of Computation, 31 (1977), pp. 954–961.

[238] ———, *Towards an efficient sparsity exploiting Newton method for minimization*, in Sparse Matrices and Their Uses, Academic Press, New York, 1981, pp. 57–87.

[239] ———, *On large-scale nonlinear least squares calculations*, SIAM Journal on Scientific and Statistical Computing, 8 (1987), pp. 416–435.

[240] L. VANDENBERGHE AND S. BOYD, *Semidefinite programming*, SIAM Review, 38 (1996), pp. 49–95.

[241] S. A. VAVASIS, *Nonlinear Optimization*, Oxford University Press, New York and Oxford, 1991.

[242] H. WALKER, *Implementation of the GMRES method using Householder transformations*, SIAM Journal on Scientific and Statistical Computing, 9 (1989), pp. 815–825.

[243] WATERLOO MAPLE SOFTWARE, INC, *Maple V software package*, 1994.

[244] L. T. WATSON, *Numerical linear algebra aspects of globally convergent homotopy methods*, SIAM Review, 28 (1986), pp. 529–545.

[245] R. B. WILSON, *A simplicial algorithm for concave programming*, Ph.D. thesis, Graduate School of Business Administration, Harvard University, 1963.

[246] W. L. WINSTON, *Operations Research*, Wadsworth Publishing Co., 3rd ed., 1997.

[247] P. WOLFE, *The composite simplex algorithm*, SIAM Review, 7 (1965), pp. 42–54.

[248] S. WOLFRAM, *The Mathematica Book*, Cambridge University Press and Wolfram Media, Inc., third ed., 1996.

[249] L. A. WOLSEY, *Integer Programming*, Wiley–Interscience Series in Discrete Mathematics and Optimization, John Wiley & Sons, New York, NY, 1998.

[250] M. H. WRIGHT, *Numerical Methods for Nonlinearly Constrained Optimization*, Ph.D. thesis, Stanford University, Stanford University, CA, 1976.

[251] ———, *Interior methods for constrained optimization*, in Acta Numerica 1992, Cambridge University Press, 1992, pp. 341–407.

[252] ———, *Ill-conditioning and computational error in interior methods for nonlinear programming*, SIAM Journal on Optimization, 9 (1999), pp. 84–111.

[253] S. J. WRIGHT, *Applying new optimization algorithms to model predictive control*, in Chemical Process Control-V, J. C. Kantor, ed., CACHE, 1997.

[254] ———, *On the convergence of the Newton/log-barrier method*, Preprint ANL/MCS-P681-0897, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, Ill., August 1997.

[255] ———, *Primal-Dual Interior-Point Methods*, SIAM Publications, Philadelphia, Pa, 1997.

[256] ———, *Effects of finite-precision arithmetic on interior-point methods for nonlinear programming*, Preprint MCS–P705–0198, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, Ill., January 1998.

[257] S. J. WRIGHT AND J. N. HOLT, *An inexact Levenberg-Marquardt method for large sparse nonlinear least squares problems*, Journal of the Australian Mathematical Society, Series B, 26 (1985), pp. 387–403.

[258] S. J. WRIGHT AND F. JARRE, *The role of linear objective functions in barrier methods*, Preprint MCS-P485-1294, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, Ill., 1994. Revised 1998. To appear in Mathematical Programming, Series A.

[259] C. ZHU, R. H. BYRD, P. LU, AND J. NOCEDAL, *Algorithm 778: L-BFGS-B, FORTRAN subroutines for large scale bound constrained optimization*, ACM Transactions on Mathematical Software, 23 (1997), pp. 550–560.

Index