Appendices

A

Notation and Definitions

All notation used in this work is "standard", and I have endeavored to use notation consistently. I have opted for simple notation, which, of course, results in a one-to-many map of notation to object classes. Within a given context, however, the overloaded notation is generally unambiguous.

This appendix is not intended to be a comprehensive listing of definitions. The Index is a more reliable set of pointers to definitions, except for symbols that are not words.

A.1 General Notation

Uppercase italic Latin and Greek letters, such as A, B, E, Λ, etc., are generally used to represent either matrices or random variables. Random variables are usually denoted by letters nearer the end of the Latin alphabet, such X, Y, and Z, and by the Greek letter E. Parameters in models (that is, unobservables in the models), whether or not they are considered to be random variables, are generally represented by lowercase Greek letters. Uppercase Latin and Greek letters are also used to represent cumulative distribution functions. Also, uppercase Latin letters are used to denote sets.

Lowercase Latin and Greek letters are used to represent ordinary scalar or vector variables and functions. **No distinction in the notation is made between scalars and vectors**; thus, β may represent a vector and β_i may represent the i^{th} element of the vector β. In another context, however, β may represent a scalar. All vectors are considered to be column vectors, although we may write a vector as $x = (x_1, x_2, \ldots, x_n)$. Transposition of a vector or a matrix is denoted by the superscript "T".

Uppercase calligraphic Latin letters, such as $\mathcal{D}$, $\mathcal{V}$, and $\mathcal{W}$, are generally used to represent either vector spaces or transforms (functionals).

© Springer International Publishing AG 2017

J.E. Gentle, *Matrix Algebra*, Springer Texts in Statistics,
DOI 10.1007/978-3-319-64867-5

Subscripts generally represent indexes to a larger structure; for example, x_{ij} may represent the $(i,j)^{\text{th}}$ element of a matrix, X. A subscript in parentheses represents an order statistic. A superscript in parentheses represents an iteration; for example, $x_i^{(k)}$ may represent the value of x_i at the k^{th} step of an iterative process.

x_i The i^{th} element of a structure (including a sample, which is a multiset).

$x_{(i)}$ The i^{th} order statistic.

$x^{(i)}$ The value of x at the i^{th} iteration.

Realizations of random variables and placeholders in functions associated with random variables are usually represented by lowercase letters corresponding to the uppercase letters; thus, ϵ may represent a realization of the random variable E.

A single symbol in an italic font is used to represent a single variable. A Roman font or a special font is often used to represent a standard operator or a standard mathematical structure. Sometimes a string of symbols in a Roman font is used to represent an operator (or a standard function); for example, $\exp(\cdot)$ represents the exponential function. But a string of symbols in an italic font on the same baseline should be interpreted as representing a composition (probably by multiplication) of separate objects; for example, *exp* represents the product of e, x, and p. Likewise a string of symbols in a Roman font (usually a single symbol) is used to represent a fundamental constant; for example, e represents the base of the natural logarithm, while e represents a variable.

A fixed-width font is used to represent computer input or output, for example,

```
a = bx + sin(c).
```

In computer text, a string of letters or numerals with no intervening spaces or other characters, such as `bx` above, represents a single object, and there is no distinction in the font to indicate the type of object.

Some important mathematical structures and other objects are:

$\mathbb{R}$ The field of reals or the set over which that field is defined.

$\mathbb{R}_+$ The set of positive reals.

$\bar{\mathbb{R}}_+$ The nonnegative reals; $\bar{\mathbb{R}}_+ = \mathbb{R}_+ \cup \{0\}$.

$\mathbb{R}^d$	The usual d-dimensional vector space over the reals or the set of all d-tuples with elements in $\mathbb{R}$.
$\mathbb{R}^{n\times m}$	The vector space of real $n \times m$ matrices.
$\mathbb{Z}$	The ring of integers or the set over which that ring is defined.
$\mathcal{G}L(n)$	The general linear group; that is, the group of $n \times n$ full rank (real) matrices with Cayley multiplication.
$\mathcal{O}(n)$	The orthogonal group; that is, the group of $n \times n$ orthogonal (orthonormal) matrices with Cayley multiplication.
e	The base of the natural logarithm. This is a constant; e may be used to represent a variable. (Note the difference in the font.)
i	The imaginary unit, $\sqrt{-1}$. This is a constant; i may be used to represent a variable. (Note the difference in the font.)

A.2 Computer Number Systems

Computer number systems are used to simulate the more commonly used number systems. It is important to realize that they have different properties, however. Some notation for computer number systems follows.

$\mathbb{F}$	The set of floating-point numbers with a given precision, on a given computer system, or this set together with the four operators `+`, `-`, `*`, and `/`. ($\mathbb{F}$ is similar to $\mathbb{R}$ in some useful ways; see Sect. 10.1.2 and Table 10.4 on page 492.)
$\mathbb{I}$	The set of fixed-point numbers with a given length, on a given computer system, or this set together with the four operators `+`, `-`, `*`, and `/`. ($\mathbb{I}$ is similar to $\mathbb{Z}$ in some useful ways; see Sect. 10.1.1.)
$e_{\min}$ and $e_{\max}$	The minimum and maximum values of the exponent in the set of floating-point numbers with a given length (see page 470).
$\epsilon_{\min}$ and $\epsilon_{\max}$	The minimum and maximum spacings around 1 in the set of floating-point numbers with a given length (see page 472).

ϵ or ϵ_{mach} The machine epsilon, the same as ϵ_{min} (see page 472).

$[\cdot]_c$ The computer version of the object $\cdot$ (see page 484).

NA Not available; a missing-value indicator.

NaN Not-a-number (see page 475).

A.3 General Mathematical Functions and Operators

Functions such as sin, max, span, and so on that are commonly associated with groups of Latin letters are generally represented by those letters in a Roman font.

Operators such as d (the differential operator) that are commonly associated with a Latin letter are generally represented by that letter in a Roman font.

Note that some symbols, such as $|\cdot|$, are overloaded; such symbols are generally listed together below.

$\times$ Cartesian or cross product of sets, or multiplication of elements of a field or ring.

$|x|$ The modulus of the real or complex number x; if x is real, $|x|$ is the absolute value of x.

$\lceil x \rceil$ The ceiling function evaluated at the real number x: $\lceil x \rceil$ is the smallest integer greater than or equal to x. For any x, $\lfloor x \rfloor \leq x \leq \lceil x \rceil$.

$\lfloor x \rfloor$ The floor function evaluated at the real number x: $\lfloor x \rfloor$ is the largest integer less than or equal to x.

$x!$ The factorial of x. If x is a positive integer, $x! = x(x-1)\cdots 2 \cdot 1$. For all other values except negative integers, $x!$ is defined by $x! = \Gamma(x+1)$.

$\mathrm{O}(f(n))$ The order class big O with respect to $f(n)$.

$$g(n) \in \mathrm{O}(f(n))$$

means there exists some fixed c such that $\|g(n)\| \leq c\|f(n)\| \; \forall n$. In particular, $g(n) \in \mathrm{O}(1)$ means $g(n)$ is bounded.
In one special case, we will use $\mathrm{O}(f(n))$ to represent some unspecified scalar or vector $x \in \mathrm{O}(f(n))$. This is the case of a convergent series. An example is

$$s = f_1(n) + \cdots + f_k(n) + \mathrm{O}(f(n)),$$

where $f_1(n), \ldots, f_k(n)$ are finite constants.
We may also express the order class defined by convergence as $x \to a$ as $\mathrm{O}(f(x))_{x \to a}$ (where a may be infinite). Hence, $g \in \mathrm{O}(f(x))_{x \to a}$ iff

$$\limsup_{x \to a} \|g(n)\|/\|f(n)\| < \infty.$$

$\mathrm{o}(f(n))$ Little o; $g(n) \in \mathrm{o}(f(n))$ means for all $c > 0$ there exists some fixed N such that $0 \leq g(n) < cf(n) \; \forall n \geq N$. (The functions f and g and the constant c could all also be negative, with a reversal of the inequalities.) Hence, $g(n) \in \mathrm{o}(f(n))$ means $\|g(n)\|/\|f(n)\| \to 0$ as $n \to \infty$.
In particular, $g(n) \in \mathrm{o}(1)$ means $g(n) \to 0$.
We also use $\mathrm{o}(f(n))$ to represent some unspecified scalar or vector $x \in \mathrm{o}(f(n))$ in special case of a convergent series, as above:

$$s = f_1(n) + \cdots + f_k(n) + \mathrm{o}(f(n)).$$

We may also express this kind of convergence in the form $g \in \mathrm{o}(f(x))_{x \to a}$ as $x \to a$ (where a may be infinite).

$\mathrm{O}_P(f(n))$ Bounded convergence in probability; $X(n) \in \mathrm{O}_P(f(n))$ means that for any positive ϵ, there is a constant C_ϵ such that $\sup_n \Pr(\|X(n)\| \geq C_\epsilon \|f(n)\|) < \epsilon$.

$\mathrm{o}_P(f(n))$ Convergent in probability; $X(n) \in \mathrm{o}_P(f(n))$ means that for any positive ϵ, $\Pr(\|X(n) - f(n)\| > \epsilon) \to 0$ as $n \to \infty$.

d The differential operator.

Δ or δ A perturbation operator; Δx (or δx) represents a perturbation of x and not a multiplication of x by Δ (or δ), even if x is a type of object for which a multiplication is defined.

$\Delta(\cdot,\cdot)$ A real-valued difference function; $\Delta(x,y)$ is a measure of the difference of x and y. For simple objects, $\Delta(x,y) = |x-y|$. For more complicated objects, a subtraction operator may not be defined, and Δ is a generalized difference.

$\tilde{x}$ A perturbation of the object x; $\Delta(x,\tilde{x}) = \Delta x$.

$\tilde{x}$ An average of a sample of objects generically denoted by x.

$\bar{x}$ The mean of a sample of objects generically denoted by x.

$\bar{x}$ The complex conjugate of the complex number x; that is, if $x = r + \mathrm{i}c$, then $\bar{x} = r - \mathrm{i}c$.

$\mathrm{sign}(x)$ For the vector x, a vector of units corresponding to the signs:

$$\begin{aligned} \mathrm{sign}(x)_i &= 1 \quad \text{if } x_i > 0, \\ &= 0 \quad \text{if } x_i = 0, \\ &= -1 \quad \text{if } x_i < 0, \end{aligned}$$

with a similar meaning for a scalar.

A.3.1 Special Functions

A good general reference on special functions in mathematics is the *NIST Handbook of Mathematical Functions*, edited by Olver et al. (2010). Another good reference on special functions is the venerable book edited by Abramowitz and Stegun (1964), which has been kept in print by Dover Publications.

$\log x$ The natural logarithm evaluated at x.

$\sin x$ The sine evaluated at x (in radians) and similarly for other trigonometric functions.

$\Gamma(x)$ The complete gamma function: $\Gamma(x) = \int_0^\infty t^{x-1}\mathrm{e}^{-t}\mathrm{d}t$. (This is called Euler's integral.) Integration by parts immediately gives the replication formula $\Gamma(x+1) = x\Gamma(x)$, and so if x is a positive integer, $\Gamma(x+1) = x!$, and more generally, $\Gamma(x+1)$ defines $x!$ for all x except negative integers. Direct evaluation of the integral yields $\Gamma(1/2) = \sqrt{\pi}$. Using this and the replication formula, with some manipulation we get for the positive integer j

$$\Gamma(j+1/2) = \frac{1 \cdot 2 \cdots (2j-1)}{2^j}\sqrt{\pi}.$$

The integral does not exist, and thus, the gamma function is not defined at the nonpositive integers.
The notation $\Gamma_d(x)$ denotes the multivariate gamma function (page 221), although in other literature this notation denotes the incomplete univariate gamma function, $\int_0^d t^{x-1}\mathrm{e}^{-t}\mathrm{d}t$.

A.4 Linear Spaces and Matrices

$\mathcal{V}(G)$ For the set of vectors (all of the same order) G, the vector space generated by that set.

$\mathcal{V}(X)$ For the matrix X, the vector space generated by the columns of X.

$\dim(\mathcal{V})$ The dimension of the vector space $\mathcal{V}$; that is, the maximum number of linearly independent vectors in the vector space.

$\mathrm{span}(Y)$ For Y either a set of vectors or a matrix, the vector space $\mathcal{V}(Y)$.

$\perp$ Orthogonality relationship (vectors, see page 33; vector spaces, see page 34).

$\mathcal{V}^\perp$ The orthogonal complement of the vector space $\mathcal{V}$ (see page 34).

$\mathcal{N}(A)$ The null space of the matrix A; that is, the set of vectors generated by all solutions, z, of the homogeneous system $Az = 0$; $\mathcal{N}(A)$ is the orthogonal complement of $\mathcal{V}(A^{\mathrm{T}})$.

$\mathrm{tr}(A)$ The trace of the square matrix A, that is, the sum of the diagonal elements.

$\mathrm{rank}(A)$ The rank of the matrix A, that is, the maximum number of independent rows (or columns) of A.

$\sigma(A)$	The spectrum of the matrix A (the set of (unique) eigenvalues).
$\rho(A)$	The spectral radius of the matrix A (the maximum absolute value of its eigenvalues).
$A > 0$ $A \geq 0$	If A is a matrix, this notation means, respectively, that each element of A is positive or nonnegative.
$A \succ 0$ $A \succeq 0$	This notation means that A is a symmetric matrix and that it is, respectively, positive definite or nonnegative definite.
A^{T}	For the matrix A, its transpose (also used for a vector to represent the corresponding row vector).
A^{H}	The conjugate transpose, also called the adjoint, of the matrix A; $A^{\mathrm{H}} = \bar{A}^{\mathrm{T}} = \overline{A^{\mathrm{T}}}$.
A^{-1}	The inverse of the square, nonsingular matrix A.
$A^{-\mathrm{R}}$	The right inverse of the $n \times m$ matrix A, of rank n; $AA^{-\mathrm{R}} = I_n$. The right inverse is $m \times n$ and of full column rank.
$A^{-\mathrm{L}}$	The left inverse of the $n \times m$ matrix A, of rank m; $A^{-\mathrm{L}}A = I_m$. The right inverse is $m \times n$ and of full row rank.
$A^{-\mathrm{T}}$	The inverse of the transpose of the square, nonsingular matrix A.
A^{+}	The g_4 inverse, the Moore-Penrose inverse, or the pseudoinverse of the matrix A (see page 128).
A^{-}	A g_1, or generalized, inverse of the matrix A (see page 128).
$A^{\frac{1}{2}}$	The square root of a nonnegative definite or positive definite matrix A; $(A^{\frac{1}{2}})^2 = A$.
$A^{-\frac{1}{2}}$	The square root of the inverse of a positive definite matrix A; $(A^{-\frac{1}{2}})^2 = A^{-1}$.

$\odot$ Hadamard multiplication (see page 94).

$\otimes$ Kronecker multiplication (see page 95).

$\oplus$ The direct sum of two matrices; $A \oplus B = \text{diag}(A, B)$ (see page 63).

$\oplus$ Direct sum of vector spaces (see page 18).

A.4.1 Norms and Inner Products

L_p For real $p \geq 1$, a norm formed by accumulating the p^{th} powers of the moduli of individual elements in an object and then taking the $(1/p)^{\text{th}}$ power of the result (see page 27).

$\|\cdot\|$ In general, the norm of the object $\cdot$.

$\|\cdot\|_p$ In general, the L_p norm of the object $\cdot$.

$\|x\|_p$ For the vector x, the L_p norm

$$\|x\|_p = \left(\sum |x_i|^p\right)^{\frac{1}{p}}$$

(see page 27).

$\|X\|_p$ For the matrix X, the L_p norm

$$\|X\|_p = \max_{\|v\|_p=1} \|Xv\|_p$$

(see page 165).

$\|X\|_{\text{F}}$ For the matrix X, the Frobenius norm

$$\|X\|_{\text{F}} = \sqrt{\sum_{i,j} x_{ij}^2}$$

(see page 167).

$\|X\|_{\mathrm{F}_p}$ For the matrix X, the Frobenius p norm

$$\|X\|_{\mathrm{F}_p} = \left(\sum_{i,j} |x_{ij}|^p\right)^{1/p}.$$

$\|X\|_{\mathrm{S}_p}$ For the $n \times m$ matrix X, the Schatten p norm

$$\|X\|_{\mathrm{S}_p} = \left(\sum_{i=1}^{\min(n,m)} d_i^p\right)^{1/p},$$

where the d_i are the singular values of X.

$\langle x, y\rangle$ The inner product or dot product of x and y (see page 23; and see page 97 for matrices).

$\kappa_p(A)$ The L_p condition number of the nonsingular square matrix A with respect to inversion (see page 269).

A.4.2 Matrix Shaping Notation

$\mathrm{diag}(A)$ or $\mathrm{vecdiag}(A)$ For the vector A, the vector consisting of the elements of the principal diagonal of A;

$$\mathrm{diag}(A) = \mathrm{vecdiag}(A) = (a_{11}, \ldots, a_{kk}),$$

where k is the minimum of the number of rows and the number of columns of A.

$\mathrm{diag}(v)$ For the vector v, the diagonal matrix whose nonzero elements are those of v; that is, the square matrix, A, such that $A_{ii} = v_i$ and for $i \neq j$, $A_{ij} = 0$.

$\mathrm{diag}(A_1, A_2, \ldots, A_k)$ The block diagonal matrix whose submatrices along the diagonal are $A_1, A_2, \ldots, A_k$.

$\mathrm{vec}(A)$ — The vector consisting of the columns of the matrix A all strung into one vector; if the column vectors of A are $a_1, a_2, \ldots, a_m$, then

$$\mathrm{vec}(A) = (a_1^{\mathrm{T}}, a_2^{\mathrm{T}}, \ldots, a_m^{\mathrm{T}}).$$

$\mathrm{vech}(A)$ — For the $m \times m$ symmetric matrix A, the vector consisting of the lower triangular elements all strung into one vector:

$$\mathrm{vech}(A) = (a_{11}, a_{21}, \ldots, a_{m1}, a_{22}, \ldots, a_{m2}, \ldots, a_{mm}).$$

$A_{(i_1,\ldots,i_k)}$ — The matrix formed from rows $i_1, \ldots, i_k$ and columns $i_1, \ldots, i_k$ from a given matrix A. This kind of submatrix and the ones below occur often when working with determinants (for square matrices). If A is square, the determinants of these submatrices are called *minors* (see page 67). Because the principal diagonal elements of this matrix are principal diagonal elements of A, it is called a principal submatrix of A. Generally, but not necessarily, $i_j < i_{j+1}$.

$A_{(i_1,\ldots,i_k)(j_1,\ldots,j_l)}$ — The submatrix of a given matrix A formed from rows $i_1, \ldots, i_k$ and columns $j_1, \ldots, j_l$ from A.

$A_{(i_1,\ldots,i_k)(*)}$ or $A_{(*)(j_1,\ldots,j_l)}$ — The submatrix of a given matrix A formed from rows $i_1, \ldots, i_k$ and all columns or else all rows and columns $j_1, \ldots, j_l$ from A.

$A_{-(i_1,\ldots,i_k)(j_1,\ldots,j_l)}$ — The submatrix formed from a given matrix A by deleting rows $i_1, \ldots, i_k$ and columns $j_1, \ldots, j_l$.

$A_{-(i_1,\ldots,i_k)()}$ or $A_{-()(j_1,\ldots,j_l)}$ — The submatrix formed from a given matrix A by deleting rows $i_1, \ldots, i_k$ (and keeping all columns) or else by deleting columns $j_1, \ldots, j_l$ from A.

A.4.3 Notation for Rows or Columns of Matrices

a_{i*} The vector that corresponds to the i^{th} row of the matrix A. As with all vectors, this is a column vector, so it often appears in the form a_{i*}^{T}.

a_{*j} The vector that corresponds to the j^{th} column of the matrix A.

A.4.4 Notation Relating to Matrix Determinants

$|A|$ The determinant of the square matrix A, $|A| = \det(A)$.

$\det(A)$ The determinant of the square matrix A, $\det(A) = |A|$.

$\det\left(A_{(i_1,\ldots,i_k)}\right)$ A principal minor of a square matrix A; in this case, it is the minor corresponding to the matrix formed from rows $i_1, \ldots, i_k$ and columns $i_1, \ldots, i_k$ from a given matrix A. The notation $|A_{(i_1,\ldots,i_k)}|$ is also used synonymously.

$\det\left(A_{-(i)(j)}\right)$ The minor associated with the $(i,j)^{\text{th}}$ element of a square matrix A. The notation $|A_{-(i)(j)}|$ is also used synonymously.

$a_{(ij)}$ The cofactor associated with the $(i,j)^{\text{th}}$ element of a square matrix A; that is, $a_{(ij)} = (-1)^{i+j}|A_{-(i)(j)}|$.

$\text{adj}(A)$ The adjugate, also called the classical adjoint, of the square matrix A: $\text{adj}(A) = (a_{(ji)})$; that is, the matrix of the same size as A formed from the cofactors of the elements of A^{T}.

A.4.5 Matrix-Vector Differentiation

$\mathrm{d}t$ The differential operator on the scalar, vector, or matrix t. This is an operator; d may be used to represent a variable. (Note the difference in the font.)

g_f or ∇f For the scalar-valued function f of a vector variable, the vector whose i^{th} element is $\partial f/\partial x_i$. This is the gradient, also often denoted as g_f.

∇f For the vector-valued function f of a vector variable, the matrix whose element in position (i, j) is

$$\frac{\partial f_j(x)}{\partial x_i}.$$

This is also written as $\partial f^{\mathrm{T}}/\partial x$ or just as $\partial f/\partial x$. This is the transpose of the Jacobian of f.

J_f For the vector-valued function f of a vector variable, the Jacobian of f denoted as J_f. The element in position (i, j) is

$$\frac{\partial f_i(x)}{\partial x_j}.$$

This is the transpose of (∇f): $\mathrm{J}_f = (\nabla f)^{\mathrm{T}}$.

H_f or $\nabla\nabla f$ or $\nabla^2 f$ The Hessian of the scalar-valued function f of a vector variable. The Hessian is the transpose of the Jacobian of the gradient. Except in pathological cases, it is symmetric. The element in position (i, j) is

$$\frac{\partial^2 f(x)}{\partial x_i \partial x_j}.$$

The symbol $\nabla^2 f$ is sometimes also used to denote the trace of the Hessian, in which case it is called the Laplace operator.

A.4.6 Special Vectors and Matrices

1 or 1_n A vector (of length n) whose elements are all 1s.

0 or 0_n A vector (of length n) whose elements are all 0s.

I or I_n The $(n \times n)$ identity matrix.

e_i The i^{th} unit vector (with implied length) (see page 16).

A.4.7 Elementary Operator Matrices

E_{pq} The $(p, q)^{\text{th}}$ elementary permutation matrix (see page 81).

$E_{(\pi)}$ The permutation matrix that permutes the rows according to the permutation π.

$E_p(a)$ The p^{th} elementary scalar multiplication matrix (see page 82).

$E_{pq}(a)$ The $(p,q)^{\text{th}}$ elementary axpy matrix (see page 83).

A.5 Models and Data

A form of model used often in statistics and applied mathematics has three parts: a left-hand side representing an object of primary interest; a function of another variable and a parameter, each of which is likely to be a vector; and an adjustment term to make the right-hand side equal the left-hand side. The notation varies depending on the meaning of the terms. One of the most common models used in statistics, the linear regression model with normal errors, is written as

$$Y = \beta^{\mathrm{T}} x + E. \tag{A.1}$$

The adjustment term is a random variable, denoted by an uppercase epsilon. The term on the left-hand side is also a random variable. This model does not represent observations or data. A slightly more general form is

$$Y = f(x;\theta) + E. \tag{A.2}$$

A single observation or a single data item that corresponds to model (A.1) may be written as

$$y = \beta^{\mathrm{T}} x + \epsilon,$$

or, if it is one of several,

$$y_i = \beta^{\mathrm{T}} x_i + \epsilon_i.$$

Similar expressions are used for a single data item that corresponds to model (A.2).

In these cases, rather than being a random variable, ϵ or ϵ_i may be a realization of a random variable, or it may just be an adjustment factor with no assumptions about its origin.

A set of n such observations is usually represented in an n-vector y, a matrix X with n rows, and an n-vector ϵ:

$$y = X\beta + \epsilon$$

or

$$y = f(X;\theta) + \epsilon.$$

B

Solutions and Hints for Selected Exercises

Exercises Beginning on Page 52

2.3b. Let one vector space consist of all vectors of the form $(a, 0)$ and the other consist of all vectors of the form $(0, b)$. The vector (a, b) is not in the union if $a \neq 0$ and $b \neq 0$.

2.8. Give a counterexample to the triangle inequality; for example, let $x = (9, 25)$ and $y = (16, 144)$.

2.11a. We first observe that if $\|x\|_p = 0$ or $\|y\|_q = 0$, we have $x = 0$ or $y = 0$, and so the inequality is satisfied because both sides are 0; hence, we need only consider the case $\|x\|_p > 0$ and $\|y\|_q > 0$. We also observe that if $p = 1$ or $q = 1$, we have the Manhattan and Chebyshev norms and the inequality is satisfied; hence we need only consider the case $1 < p < \infty$.

Now, for p and q as given, for any numbers a_i and b_i, there are numbers s_i and t_i such that $|a_i| = \mathrm{e}^{s_i/p}$ and $|b_i| = \mathrm{e}^{t_i/q}$. Because e^x is a convex function, we have $\mathrm{e}^{s_i/p+t_i/q} \leq \frac{1}{p}\mathrm{e}^s_i + \frac{1}{q}\mathrm{e}^t_i$, or

$$a_i b_i \leq |a_i||b_i| \leq |a_i|^p/p + |b_i|^q/q.$$

Now let

$$a_i = \frac{x_i}{\|x\|_p} \quad \text{and} \quad b_i = \frac{y_i}{\|y\|_q},$$

and so

$$\frac{x_i}{\|x\|_p}\frac{y_i}{\|y\|_q} \leq \frac{1}{p}\frac{|x_i|^p}{\|x\|_p^p} + \frac{1}{q}\frac{|y_i|^q}{\|y\|_q^q}.$$

© Springer International Publishing AG 2017

J.E. Gentle, *Matrix Algebra*, Springer Texts in Statistics,
DOI 10.1007/978-3-319-64867-5

Now, summing these equations over i, we have

$$\begin{aligned}\frac{\langle x, y\rangle}{\|x\|_p\|y\|_q} &\leq \frac{1}{p}\frac{\|x\|_p^p}{\|x\|_p^p} + \frac{1}{q}\frac{\|y\|_q^q}{\|y\|_q^q} \\ &= 1.\end{aligned}$$

Hence, we have the desired result.
As we see from this proof, the inequality is actually a little stronger than stated. If we define u and v by $u_i = |x_i|$ and $v_i = |y_i|$, we have

$$\langle x, y\rangle \leq \langle u, v\rangle \leq \|x\|_p\|y\|_q.$$

We observe that equality occurs if and only if

$$\left(\frac{|x_i|}{\|x\|_p}\right)^{\frac{1}{q}} = \left(\frac{|y_i|}{\|y\|_q}\right)^{\frac{1}{p}}$$

and

$$\operatorname{sign}(x_i) = \operatorname{sign}(y_i)$$

for all i.
We note a special case by letting $y = 1$:

$$\bar{x} \leq \|x\|_p,$$

and with $p = 2$, we have a special case of the Cauchy-Schwarz inequality,

$$n\bar{x}^2 \leq \|x\|_2^2,$$

which guarantees that $\mathrm{V}(x) \geq 0$.

2.11b. Using the triangle inequality for the absolute value, we have $|x_i + y_i| \leq |x_i| + |y_i|$. This yields the result for $p = 1$ and $p = \infty$ (in the limit). Now assume $1 < p < \infty$. We have

$$\|x + y\|_p^p \leq \sum_{i=1}^n |x_i + y_i|^{p-1}|x_i| + \sum_{i=1}^n |x_i + y_i|^{p-1}|y_i|.$$

Now, letting $q = p/(p-1)$, we apply Hölder's inequality to each of the terms on the right:

$$\sum_{i=1}^n |x_i + y_i|^{p-1}|x_i| \leq \left(\sum_{i=1}^n |x_i + y_i|^{(p-1)q}\right)^{\frac{1}{q}} \left(\sum_{i=1}^n |x_i|^p\right)^{\frac{1}{p}}$$

and

$$\sum_{i=1}^n |x_i + y_i|^{p-1}|x_i| \leq \left(\sum_{i=1}^n |x_i + y_i|^{(p-1)q}\right)^{\frac{1}{q}} \left(\sum_{i=1}^n |y_i|^p\right)^{\frac{1}{p}},$$

so

$$\sum_{i=1}^{n}|x_i+y_i|^p \le \left(\sum_{i=1}^{n}|x_i+y_i|^{(p-1)q}\right)^{\frac{1}{q}}\left(\left(\sum_{i=1}^{n}|x_i|^p\right)^{\frac{1}{p}}+\left(\sum_{i=1}^{n}|y_i|^p\right)^{\frac{1}{p}}\right)$$

or, because $(p-1)q = p$ and $1-\frac{1}{q}=\frac{1}{p}$,

$$\left(\sum_{i=1}^{n}|x_i+y_i|^p\right)^{\frac{1}{p}} \le \left(\sum_{i=1}^{n}|x_i|^p\right)^{\frac{1}{p}}+\left(\sum_{i=1}^{n}|y_i|^p\right)^{\frac{1}{p}},$$

which is the same as

$$\|x+y\|_p \le \|x\|_p + \|y\|_p,$$

the triangle inequality.

2.19e. In $\mathbb{R}^3$,

$$\text{angle}(x,y) = \sin^{-1}\left(\frac{\|x\times y\|}{\|x\|\,\|y|}\right).$$

Because $x\times y = -y\times x$, this allows us to determine the angle from x to y; that is, the *direction* within $(-\pi,\pi]$ in which x would be rotated to y.

2.21. Just consider the orthogonal vectors $x=(1,0)$ and $y=(0,1)$. The centered vectors are $x_c=(\frac{1}{2},-\frac{1}{2})$ and $y_c=(-\frac{1}{2},\frac{1}{2})$. The angle between the uncentered vectors is $\pi/2$, while that between the centered vectors is π.

Exercises Beginning on Page 178

3.2. This exercise occurs in various guises in many different places, and the simple approach is to add and subtract $\bar{x}$:

$$\begin{aligned}(x-a)^{\mathrm{T}}(x-a) &= (x-\bar{x}-(a+\bar{x}))^{\mathrm{T}}(x-\bar{x}-(a+\bar{x}))\\ &= (x_c-(a+\bar{x}))^{\mathrm{T}}(x_c-(a+\bar{x}))\\ &= x_c^{\mathrm{T}}x_c+(a+\bar{x})^{\mathrm{T}}(a+\bar{x}))-2(a+\bar{x})^{\mathrm{T}}x_c\\ &= x_c^{\mathrm{T}}x_c+n(a+\bar{x})^2.\end{aligned}$$

Finally, we get the expressions in equation (3.92) by writing $x_c^{\mathrm{T}}x_c$ as $\mathrm{tr}(x_cx_c^{\mathrm{T}})$.

3.14. Write the $n\times m$ matrix A as

$$A=(a_{ij})=[a_{*1},\ldots,a_{*m}].$$

If A is of rank one, the maximum number of linearly independent columns is one; hence, for $k=2,\ldots,m$, $a_{*k}=c_ka_{*1}$, for some c_k.

Now let $x = a_{*1}$, which is an n-vector, and let y be an m-vector whose first element is 1 and whose $k = 2, \ldots, m$ elements are the c_ks. We see that $A = xy^{\mathrm{T}}$ by direct multiplication.
This decomposition is not unique, of course.

3.15. If the elements of the square matrix A are integers then each cofactor $a_{(ij)}$ is an integer, and hence the elements of $\mathrm{adj}(A)$ are integers. If $|A| = \pm 1$, then $A^{-1} = \pm\mathrm{adj}(A)$, and so the elements of A^{-1} are integers.
An easy way to form a matrix whose determinant is ± 1 is to form an upper triangular matrix with either 1 or -1 on the diagonal. A more "interesting matrix" that has the same determinant can then be formed by use of elementary operations. (People teaching matrix algebra find this useful!)

3.16. Because the inverse of a matrix is unique, we can verify each equation by multiplication by the inverse at the appropriate place. We can often reduce an expression to a desired form by multiplication by the identity MM^{-1}, where M is some appropriate matrix. For example, equation (3.177) can be verified by the equations

$$\begin{aligned}
(A+B)(A^{-1} - A^{-1}(A^{-1}+B^{-1})^{-1}A^{-1}) &= \\
I - (A^{-1}+B^{-1})^{-1}A^{-1} + BA^{-1} - BA^{-1}(A^{-1}+B^{-1})^{-1}A^{-1} &= \\
I + BA^{-1} - (I+BA^{-1})(A^{-1}+B^{-1})^{-1}A^{-1} &= \\
(I+BA^{-1})(I - (A^{-1}+B^{-1})^{-1}A^{-1}) &= \\
B(B^{-1}+A^{-1})(I - (A^{-1}+B^{-1})^{-1}A^{-1}) &= \\
B(B^{-1}+A^{-1}) - BA^{-1} &= I
\end{aligned}$$

3.19. Express the nonzero elements of row i in the $n \times n$ matrix A as $a_i b_k$ for $k = i, \ldots, n$, and the nonzero elements of column j as $a_k b_j$ for $k = j, \ldots, n$. Then obtain expressions for the elements of A^{-1}. Show, for example, that the diagonal elements of A^{-1} are $(a_i b_i)^{-1}$.

3.25. For property 8, let c be a nonzero eigenvalue of AB. Then there exists v ($\neq 0$) such that $ABv = cv$, that is, $BABv = Bcv$. But this means $BAw = cw$, where $w = Bv \neq 0$ (because $ABv \neq 0$) and so c is an eigenvalue of BA. We use the same argument starting with an eigenvalue of BA. For square matrices, there are no other eigenvalues, so the set of eigenvalues is the same.
For property 9, see the discussion of similarity transformations on page 146.

3.37. Let A and B be such that AB is defined.

$$\|AB\|_{\mathrm{F}}^2 = \sum_{ij} \left| \sum_k a_{ik} b_{kj} \right|^2$$

$$\begin{aligned}
&\leq \sum_{ij}\left(\sum_{k} a_{ik}^2\right)\left(\sum_{k} b_{kj}^2\right) \quad \text{(Cauchy-Schwarz)} \\
&= \left(\sum_{i,k} a_{ik}^2\right)\left(\sum_{k,j} b_{kj}^2\right) \\
&= \|A\|_{\mathrm{F}}^2 \|B\|_{\mathrm{F}}^2.
\end{aligned}$$

3.40. Hints.
For inequality (3.307), use the Cauchy-Schwartz inequality.
For inequality (3.309), use Hölder's inequality.

Exercises Beginning on Page 222

4.7b. The first step is to use the trick of equation (3.90), $x^{\mathrm{T}}Ax = \mathrm{tr}(Axx^{\mathrm{T}})$, again to undo the earlier expression, and write the last term in equation (4.44) as

$$-\frac{n}{2}\mathrm{tr}\left(\Sigma^{-1}(\bar{y}-\mu)(\bar{y}-\mu)^{\mathrm{T}}\right) = -\frac{n}{2}(\bar{y}-\mu)\Sigma^{-1}(\bar{y}-\mu)^{\mathrm{T}}.$$

Now Σ^{-1} is positive definite, so $(\bar{y}-\mu)\Sigma^{-1}(\bar{y}-\mu)^{\mathrm{T}} \geq 0$ and hence is minimized for $\hat{\mu} = \bar{y}$. Decreasing this term increases the value of $l(\mu, \Sigma; y)$, and so $l(\hat{\mu}, \Sigma; y) \geq l(\mu, \Sigma; y)$ for all positive definite Σ^{-1}.
Now, we consider the other term. Let $A = \sum_{i=1}^{n}(y_i - \bar{y})(y_i - \bar{y})^{\mathrm{T}}$. The first question is whether A is positive definite. We will refer to a text on multivariate statistics for the proof that A is positive definite with probability 1 (see Muirhead, 1982, for example). We have

$$\begin{aligned}
l(\hat{\mu}, \Sigma; y) &= c - \frac{n}{2}\log|\Sigma| - \frac{1}{2}\mathrm{tr}\left(\Sigma^{-1}A\right) \\
&= c - \frac{n}{2}\left(\log|\Sigma| + \mathrm{tr}\left(\Sigma^{-1}A/n\right)\right).
\end{aligned}$$

Because c is constant, the function is maximized at the minimum of the latter term subject to Σ being positive definite, which, as shown for expression (4.61), occurs at $\widehat{\Sigma} = A/n$.

4.12. $2^{dn/2}\Gamma_d(n/2)|\Sigma|^{n/2}$.
Make the change of variables $W = 2\Sigma^{\frac{1}{2}}Y\Sigma^{\frac{1}{2}}$, determine the Jacobian, and integrate.

Exercises Beginning on Page 261

5.2. The R code that will produce the graph is

```
x<-c(0,1)
y<-c(0,1)
z<-matrix(c(0,0,1,1),nrow=2)
trans<-persp(x, y, z, theta = 45, phi = 30)
bottom<-c(.5,0,0,1)%*%trans
top<-c(.5,1,1,1)%*%trans
xends<-c(top[,1]/top[,4],bottom[,1]/bottom[,4])
yends<-c(top[,2]/top[,4],bottom[,2]/bottom[,4])
lines(xends,yends,lwd=2)
```

5.5. Let A is an $n \times n$ matrix, whose columns are the same as the vectors a_j, and let QR be the QR factorization of A. Because Q is orthogonal, $\det(Q) = 1$, and $\det(R) = \det(A)$. Hence, we have

$$\begin{aligned} |\det(A)| &= |\det(R)| \\ &= \prod_{j=1}^{n} |r_{jj}| \\ &\leq \prod_{j=1}^{n} \|r_j\|_2 \\ &= \prod_{j=1}^{n} \|a_j\|_2, \end{aligned}$$

where r_j is the vector whose elements are the same as the elements in the j^{th} column of R.
If equality holds, then either some a_j is zero, or else $r_{jj} = \|r_j\|$ for $j = 1, \ldots, n$. In the latter case, R is diagonal, and hence $A^{\mathrm{T}}A$ is diagonal, and so the columns of A are orthogonal.

Exercises Beginning on Page 305

6.1. First, show that

$$\max_{x \neq 0} \frac{\|Ax\|}{\|x\|} = \left(\min_{x \neq 0} \frac{\|A^{-1}x\|}{\|x\|} \right)^{-1}$$

and

$$\max_{x \neq 0} \frac{\|A^{-1}x\|}{\|x\|} = \left(\min_{x \neq 0} \frac{\|Ax\|}{\|x\|} \right)^{-1}.$$

6.2a. The matrix prior to the first elimination is

$$\begin{bmatrix} 2 & 5 & 3 & 19 \\ 1 & 4 & 1 & 12 \\ 1 & 2 & 2 & 9 \end{bmatrix}.$$

The solution is $(3, 2, 1)$.

6.2b. The matrix prior to the first elimination is

$$\begin{bmatrix} 5 & 2 & 3 & 19 \\ 4 & 1 & 1 & 12 \\ 2 & 1 & 2 & 9 \end{bmatrix},$$

and x_1 and x_2 have been interchanged.

6.2c.

$$D = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 2 \end{bmatrix},$$

$$L = \begin{bmatrix} 0 & 0 & 0 \\ 2 & 0 & 0 \\ 1 & 2 & 0 \end{bmatrix},$$

$$U = \begin{bmatrix} 0 & 4 & 1 \\ 0 & 0 & 3 \\ 0 & 0 & 0 \end{bmatrix},$$

$$\rho((D + L)^{-1}U) = 1.50.$$

6.2e. $\rho((\widetilde{D} + \widetilde{L})^{-1}\widetilde{U}) = 0.9045$.

6.2g. Some R code for this is

```
tildeD <- matrix(c(2, 0, 0,
                   0, 4, 0,
                   0, 0, 2), nrow=3, byrow=T)
tildeL <- matrix(c(0, 0, 0,
                   1, 0, 0,
                   1, 2, 0), nrow=3, byrow=T)
tildeU <- matrix(c(0, 5, 3,
                   0, 0, 1,
                   0, 0, 0), nrow=3, byrow=T)
b <- c(12,19,9)

omega <- 0.1
tildeDUadj <- (1-omega)*tildeD - omega*tildeU
tildeAk <- tildeD+omega*tildeL
badj <- omega*b
xk <- c(1,1,1)
```

```
nstep <- 2
for (i in 1:nstep){
   bk <- tildeDUadj%*%xk + badj
   xkp1 <- solve(tildeAk,bk)
   dif <- sqrt(sum((xkp1-xk)^2))
   print(dif)
   xk <- xkp1
}
```

6.4a. $nm(m+1) - m(m+1)/2$. (Remember $A^{\mathrm{T}}A$ is symmetric.)

6.4g. Using the normal equations with the Cholesky decomposition requires only about half as many flops as the QR, when n is much larger than m. The QR method oftens yields better accuracy, however.

6.5a.
1. $X(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}X = X$
2. $(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}X(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}} = (X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}$
3. $X(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}$ is symmetric (take its transpose).
4. $(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}X$ is symmetric.

Therefore, $(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}} = X^{+}$.

6.5b. We want to show $X^{\mathrm{T}}(y - XX^{+}y) = 0$. Using the properties of X^{+}, we have

$$\begin{aligned} X^{\mathrm{T}}(y - XX^{+}y) &= X^{\mathrm{T}}y - X^{\mathrm{T}}XX^{+}y \\ &= X^{\mathrm{T}}y - X^{\mathrm{T}}(XX^{+})^{\mathrm{T}}y \quad \text{because of symmetry} \\ &= X^{\mathrm{T}}y - X^{\mathrm{T}}(X^{+})^{\mathrm{T}}X^{\mathrm{T}}y \\ &= X^{\mathrm{T}}y - X^{\mathrm{T}}(X^{\mathrm{T}})^{+}X^{\mathrm{T}}y \quad \text{property of Moore-Penrose inverses and transposes} \\ &= X^{\mathrm{T}}y - X^{\mathrm{T}}y \quad \text{property of Moore-Penrose inverses} \\ &= 0 \end{aligned}$$

Exercises Beginning on Page 324

7.1a. 1.

7.1b. 1.

7.1d. 1. (All that was left was to determine the probability that $c_n \neq 0$ and $c_{n-1} \neq 0$.)

7.2a. 11.6315.

7.3.

$$\begin{bmatrix} 3.08 & -0.66 & 0 & 0 \\ -0.66 & 4.92 & -3.27 & 0 \\ 0 & -3.27 & 7.00 & -3.74 \\ 0 & 0 & -3.74 & 7.00 \end{bmatrix}.$$

Exercises Beginning on Page 396

8.10a.

$$p(c) = c^m - \alpha_1 c^{m-1} - \alpha_2\sigma_1 c^{m-2} - \alpha_3\sigma_1\sigma_2 c^{m-3} - \cdots - \alpha_m\sigma_1\sigma_2\cdots\sigma_{m-1}.$$

8.10b. Define

$$f(c) = 1 - \frac{p(c)}{c^m}.$$

This is a monotone decreasing continuous function in c, with $f(c) \to \infty$ as $c \to 0_+$ and $f(c) \to 0$ as $c \to \infty$. Therefore, there is a unique value c_* for which $f(c_*) = 1$. The uniqueness also follows from Descartes' rule of signs, which states that the maximum number of positive roots of a polynomial is the number of sign changes of the coefficients, and in the case of the polynomial $p(c)$, this is one.

8.18. This is a simple case of matrix multiplication.
To illustrate the use of R in complex matrices, I will show some code that is relevant to this problem, for a given order, of course.

```
omegajn <- function(n){
#####    Function to create the n^th roots of 1  #####
omegajn <- complex(n)
omegajn[1] <- 1
if (n>=2) omegajn[2] <- complex(re=cos(2*pi/n),
im=sin(2*pi/n))
if (n>=3) for (j in 3:n) omegajn[j] <- omegajn[2]
*omegajn[j-1]
return(omegajn)
}

Fn <- function(n){
#####    Function to create a Fourier matrix  #####
rts <- omegajn(n)
Fn <- matrix(c(rep(1,n),rts),nrow=n)
if (n>=3) for (j in 3:n) Fn <- cbind(Fn,rts^(j-1))
Fn <- Fn/sqrt(n)
return(Fn)
}

# perform multiplications to get the elementary
circulant matrix or order 5
round(Conj(t(F5))%*%diag(rts5)%*%F5)
```

8.19. $(-1)^{\lfloor n/2 \rfloor} n^n$, where $\lfloor \cdot \rfloor$ is the floor function (the greatest integer function). For $n = 1, 2, 3, 4$, the determinants are $1, -4, -27, 256$.

Exercises Beginning on Page 452

9.1. 1. This is because the subspace that generates a singular matrix is a lower dimensional space than the full sample space, and so its measure is 0.

9.4d. Assuming W is positive definite, we have

$$\widehat{\beta}_{W,C} = (X^{\mathrm{T}}WX)^{-1}X^{\mathrm{T}}Wy + (X^{\mathrm{T}}WX)^{-1}L^{\mathrm{T}}(L(X^{\mathrm{T}}WX)^{+}L^{\mathrm{T}})^{-1}(c - L(X^{\mathrm{T}}WX)^{+}X^{\mathrm{T}}Wy)\,.$$

9.12. Let $X = [X_{\mathrm{i}} \mid X_{\mathrm{o}}]$ and $Z = X_{\mathrm{o}}^{\mathrm{T}}X_{\mathrm{o}} - X_{\mathrm{o}}^{\mathrm{T}}X_{\mathrm{i}}(X_{\mathrm{i}}^{\mathrm{T}}X_{\mathrm{i}})^{-1}X_{\mathrm{i}}^{\mathrm{T}}X_{\mathrm{o}}$. Note that $X_{\mathrm{o}}^{\mathrm{T}}X_{\mathrm{i}} = X_{\mathrm{i}}^{\mathrm{T}}X_{\mathrm{o}}$. We have

$$\begin{aligned}
&X_{\mathrm{i}}^{\mathrm{T}}X(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}} \\
&= X_{\mathrm{i}}^{\mathrm{T}}[X_{\mathrm{i}} \mid X_{\mathrm{o}}]\left[\begin{array}{c|c} X_{\mathrm{i}}^{\mathrm{T}}X_{\mathrm{i}} & X_{\mathrm{i}}^{\mathrm{T}}X_{\mathrm{o}} \\ X_{\mathrm{o}}^{\mathrm{T}}X_{\mathrm{i}} & X_{\mathrm{o}}^{\mathrm{T}}X_{\mathrm{o}} \end{array}\right]^{-1}[X_{\mathrm{i}} \mid X_{\mathrm{o}}]^{\mathrm{T}} \\
&= [X_{\mathrm{i}}^{\mathrm{T}}X_{\mathrm{i}} \mid X_{\mathrm{i}}^{\mathrm{T}}X_{\mathrm{o}}] \\
&\quad \left[\begin{array}{c|c} (X_{\mathrm{i}}^{\mathrm{T}}X_{\mathrm{i}})^{-1} - (X_{\mathrm{i}}^{\mathrm{T}}X_{\mathrm{i}})^{-1}(X_{\mathrm{o}}^{\mathrm{T}}X_{\mathrm{i}})Z^{-1}(X_{\mathrm{i}}^{\mathrm{T}}X_{\mathrm{o}})(X_{\mathrm{i}}^{\mathrm{T}}X_{\mathrm{i}})^{-1} & \\ -Z^{-1}(X_{\mathrm{o}}^{\mathrm{T}}X_{\mathrm{i}})(X_{\mathrm{i}}^{\mathrm{T}}X_{\mathrm{i}})^{-1} & -(X_{\mathrm{i}}^{\mathrm{T}}X_{\mathrm{i}})^{-1}(X_{\mathrm{i}}^{\mathrm{T}}X_{\mathrm{o}})Z^{-1} \\ & Z^{-1} \end{array}\right] \\
&\quad \left[\begin{array}{c} X_{\mathrm{i}}^{\mathrm{T}} \\ X_{\mathrm{o}}^{\mathrm{T}} \end{array}\right] \\
&= [I - (X_{\mathrm{o}}^{\mathrm{T}}X_{\mathrm{i}})Z^{-1}(X_{\mathrm{i}}^{\mathrm{T}}X_{\mathrm{o}})(X_{\mathrm{i}}^{\mathrm{T}}X_{\mathrm{i}})^{-1} - X_{\mathrm{i}}^{\mathrm{T}}X_{\mathrm{o}}Z^{-1}(X_{\mathrm{o}}^{\mathrm{T}}X_{\mathrm{i}})(X_{\mathrm{i}}^{\mathrm{T}}X_{\mathrm{i}})^{-1} \quad \mid \\
&\qquad\qquad -X_{\mathrm{i}}^{\mathrm{T}}X_{\mathrm{o}}Z^{-1} + X_{\mathrm{i}}^{\mathrm{T}}X_{\mathrm{o}}Z^{-1}] \\
&\quad \left[\begin{array}{c} X_{\mathrm{i}}^{\mathrm{T}} \\ X_{\mathrm{o}}^{\mathrm{T}} \end{array}\right] \\
&= X_{\mathrm{i}}^{\mathrm{T}},
\end{aligned}$$

9.14. One possibility is

$$\left[\begin{array}{cc} 20 & 100 \\ 5 & 25 \\ 5 & 25 \\ 10 & \texttt{NA} \\ 10 & \texttt{NA} \\ 10 & \texttt{NA} \\ \texttt{NA} & 10 \\ \texttt{NA} & 10 \\ \texttt{NA} & 10 \end{array}\right].$$

The variance-covariance matrix computed from all pairwise complete observations is

$$\begin{bmatrix} 30 & 375 \\ 375 & 1230 \end{bmatrix},$$

while that computed only from complete cases is

$$\begin{bmatrix} 75 & 375 \\ 375 & 1875 \end{bmatrix}.$$

The correlation matrix computed from all pairwise complete observations is

$$\begin{bmatrix} 1.00 & 1.95 \\ 1.95 & 1.00 \end{bmatrix}.$$

Note that this example is not a pseudo-correlation matrix.
In the R software system, the `cov` and `cor` functions have an argument called "`use`", which can take the values "all.obs", "complete.obs", or "pairwise.complete.obs". The value "all.obs" yields an error if the data matrix contains any missing values. In `cov`, the values "complete.obs" and "pairwise.complete.obs" yield the variance-covariances shown above. The function `cor` with `use=‘‘pairwise.complete.obs’’` yields

$$\begin{bmatrix} 1.00 & 1.00 \\ 1.00 & 1.00 \end{bmatrix}.$$

However, if `cov` is invoked with `use=‘‘pairwise.complete.obs’’` and the function `cov2cor` is applied to the result, the correlations are 1.95, as in the first correlation matrix above.

9.17. This is an open question. If you get a proof of convergence, submit it for publication. You may wish to try several examples and observe the performance of the intermediate steps. I know of no case in which the method has not converged.

9.19b. Starting with the correlation matrix given above as a possible solution for Exercise 9.14, four iterations of equation (9.67) using $\delta = 0.05$ and $f(x) = \tanh(x)$ yield

$$\begin{bmatrix} 1.00 & 0.997 \\ 0.997 & 1.00 \end{bmatrix}.$$

9.21. We can develop a recursion for p_{11}^t based on p_{11}^{t-1} and p_{12}^{t-1},

$$p_{11}^t = p_{11}^{t-1}(1-\alpha) + p_{12}^{t-1}\beta,$$

and because $p_{11} + p_{12} = 1$, we have $p_{11}^t = p_{11}^{t-1}(1-\alpha-\beta) + \beta$. Putting this together, we have

$$\lim_{t\to\infty} P = \begin{bmatrix} \beta/(\alpha+\beta) & \alpha/(\alpha+\beta) \\ \beta/(\alpha+\beta) & \alpha/(\alpha+\beta) \end{bmatrix},$$

and so the limiting (and invariant) distribution is $\pi_s = (\beta/(\alpha+\beta),\ \alpha/(\alpha+\beta))$.

9.22c. From the exponential growth, we have $N^{(T)} = N^{(0)}e^{rT}$; hence,

$$r = \frac{1}{T}\log\left(N^{(T)}/N^{(0)}\right) = \frac{1}{T}\log(r_0).$$

Exercises Beginning on Page 516

10.1a. The computations do not overflow. The first floating-point number x such that $x + 1 = x$ is

$$0.10\cdots0 \times b^{p+1}.$$

Therefore, the series converges at the value of i such that $i(i+1)/2 = x$. Now solve for i.

10.2. The function is $\log(n)$, and Euler's constant is $0.57721\ldots$.

10.6. 2^{-56}. (The standard has 53 bits normalized, so the last bit is 2^{-55}, and half of that is 2^{-56}.)

10.7a. Normalized: $2b^{p-1}(b-1)(e_{\max} - e_{\min} + 1) + 1$.
Nonnormalized: $2b^{p-1}(b-1)(e_{\max} - e_{\min} + 1) + 1 + 2b^{p-1}$.

10.7b. Normalized: $b^{e_{\min}-1}$.
Nonnormalized: $b^{e_{\min}-p}$.

10.7c. $1 + b^{-p+1}$ or $1 + b^{-p}$ when $b = 2$ and the first bit is hidden.

10.7d. b^p.

10.7e. 22.

10.12. First of all, we recognize that the full sum in each case is 1. We therefore accumulate the sum from the direction in which there are fewer terms. After computing the first term from the appropriate direction, take a logarithm to determine a scaling factor, say s^k. (This term will be the smallest in the sum.) Next, proceed to accumulate terms until the sum is of a different order of magnitude than the next term. At that point, perform a scale adjustment by dividing by s. Resume summing, making similar scale adjustments as necessary, until the limit of the summation is reached.

10.14. The result is close to 1.
What is relevant here is that numbers close to 1 have only a very few digits of accuracy; therefore, it would be better to design this program so that it returns $1 - \Pr(X \le x)$ (the "significance level"). The purpose and the anticipated use of a program determine how it should be designed.

10.17a. 2.

10.17b. 0.

10.17c. No (because the operations in the "for" loop are not chained).

10.18c.

$$
\begin{aligned}
&a = x_1\\
&b = y_1\\
&s = 0\\
&\text{for } i = 2, n\\
&\{\\
&\quad d = (x_i - a)/i\\
&\quad e = (y_i - b)/i\\
&\quad a = d + a\\
&\quad b = e + b\\
&\quad s = i(i-1)de + s\\
&\}.
\end{aligned}
$$

10.21. 1. No; 2. Yes; 3. No; 4. No.

10.22. A very simple example is

$$\begin{bmatrix} 1 & 1+\epsilon \\ 1 & 1 \end{bmatrix},$$

where $\epsilon < b^{-p}$, because in this case the matrix stored in the computer would be singular. Another example is

$$\begin{bmatrix} 1 & a(1+\epsilon) \\ a(1+\epsilon) & a^2(1+2\epsilon) \end{bmatrix},$$

where ϵ is the machine epsilon.

Exercises Beginning on Page 537

11.2a. $O(nk)$.

11.2c. At each successive stage in the fan-in, the number of processors doing the additions goes down by approximately one-half.
If $p \approx k$, then $O(n \log k)$ (fan-in on one element of c at a time)
If $p \approx nk$, then $O(\log k)$ (fan-in on all elements of c simultaneously)
If p is a fixed constant smaller than k, the order of time does not change; only the multiplicative constant changes.
Notice the difference in the order of time and the order of the number of computations. Often there is very little that can be done about the order of computations.

11.2d. Because in a serial algorithm the magnitudes of the summands become more and more different. In the fan-in, they are more likely to remain relatively equal. Adding magnitudes of different quantities results in benign roundoff, but many benign roundoffs become bad. (This is not

catastrophic cancellation.) Clearly, if all elements are nonnegative, this argument would hold. Even if the elements are randomly distributed, there is likely to be a drift in the sum (this can be thought of as a random walk). There is *no difference* in the number of computations.

11.2e. Case 1: $p \approx n$. Give each c_i a processor – do an outer loop on each. This would likely be more efficient because all processors are active at once.
Case 2: $p \approx nk$. Give each $a_{ij}b_j$ a processor – fan-in for each. This would be the same as the other.
If p is a fixed constant smaller than n, set it up as in Case 1, using n/p groups of c_i's.

11.2f. If $p \approx n$, then $O(k)$.
If $p \approx nk$, then $O(\log k)$.
If p is some small fixed constant, the order of time does not change; only the multiplicative constant changes.

Exercises Beginning on Page 582

12.2. Here is a recursive Matlab function for the Strassen algorithm due to Coleman and Van Loan. When it uses the Strassen algorithm, it requires the matrices to have even dimension.

```
   function C = strass(A,B,nmin)
%
% Strassen matrix multiplication C=AB
%       A, B must be square and of even dimension
% From Coleman and Van Loan
% If n <= nmin, the multiplication is done conventionally
%
   [n n ] = size(A);
   if n <= nmin
      C = A * B;   % n is small, get C conventionally
   else
      m = n/2; u = 1:m; v = m+1:n;
      P1 = strass(A(u,u)+A(v,v), B(u,u)+B(v,v), nmin);
      P2 = strass(A(v,u)+A(v,v), B(u,u),        nmin);
      P3 = strass(A(u,u),        B(u,v)-B(v,v), nmin);
      P4 = strass(A(v,v),        B(v,u)-B(u,u), nmin);
      P5 = strass(A(u,u)+A(u,v), B(v,v),        nmin);
      P6 = strass(A(v,u)-A(u,u), B(u,u)+B(u,v), nmin);
      P7 = strass(A(u,v)-A(v,v), B(v,u)+B(v,v), nmin);
      C = [P1+P4-P5+P7 P3+P5; P2+P4 P1+P3-P2+P6];
   end
```

12.5a.

```
real a(4,3)
data a/3.,6.,8.,2.,5.,1.,6.,3.,6.,2.,7.,1./
n = 4
m = 3
x1 = a(2,2) ! Temporary variables must be used
     because of
x2 = a(4,2) ! the side effects of srotg.
call srotg(x1, x2,, c, s)
call srot(m, a(2,1), n, a(4,1), n, c, s)
print *, c, s
print *, a
end
```

This yields 0.3162278 and 0.9486833 for c and s. The transformed matrix is

$$\begin{bmatrix} 3.000000 & 5.000000 & 6.000000 \\ 3.794733 & 3.162278 & 1.581139 \\ 8.000000 & 6.000000 & 7.000000 \\ -5.059644 & -0.00000002980232 & -1.581139 \end{bmatrix}.$$

12.7b. Using the Matrix package in R, after initializing `rho` and `sig2`, this is

```
Vinv <- sparseMatrix(i=c(1,1,2,2,2,3,3,3,4,4,4,5,5,5,6,6,6,7,
                         7,7,8,8,8,9,9,9,10,10),
                     j=c(1,2,1:3,2:4,3:5,4:6,5:7,6:8,7:9,8:10,
                         9,10),
                     x=c(1,-rho,rep(c(-rho,1+rho^2,-rho),8),
                         -rho,1))/((1-rho^2)*sig2)
```

12.9. 10.7461941829033 and 10.7461941829034.

12.11.

$$\frac{-(fh)+ei}{-(ceg)+bfg+cdh-afh-bdi+aei} \qquad \frac{ch-bi}{-(ceg)+bfg+cdh-afh-bdi+aei}$$

$$\frac{-(ce)+bf}{-(ceg)+bfg+cdh-afh-bdi+aei} \qquad \frac{fg-di}{-(ceg)+bfg+cdh-afh-bdi+aei}$$

$$\frac{-(cg)+ai}{-(ceg)+bfg+cdh-afh-bdi+aei} \qquad \frac{cd-af}{-(ceg)+bfg+cdh-afh-bdi+aei}$$

$$\frac{-(eg)+dh}{-(ceg)+bfg+cdh-afh-bdi+aei} \qquad \frac{bg-ah}{-(ceg)+bfg+cdh-afh-bdi+aei}$$

$$\frac{-(bd)+ae}{-(ceg)+bfg+cdh-afh-bdi+aei}$$

Bibliography

Many of the most useful background material is available on the internet. For statistics, one of the most useful sites on the internet is the electronic repository `statlib`, maintained at Carnegie Mellon University, which contains programs, datasets, and other items of interest. The URL is

`http://lib.stat.cmu.edu`.

The collection of algorithms published in *Applied Statistics* is available in `statlib`. These algorithms are sometimes called the *ApStat* algorithms.

Another very useful site for scientific computing is `netlib`,. The URL is

`http://www.netlib.org`

The *Collected Algorithms of the ACM* (*CALGO*), which are the Fortran, C, and Algol programs published in *ACM Transactions on Mathematical Software* (or in *Communications of the ACM* prior to 1975), are available in `netlib` under the TOMS link.

A wide range of software is used in the computational sciences. Some of the software is produced by a single individual who is happy to share the software, sometimes for a fee, but who has no interest in maintaining it. At the other extreme is software produced by large commercial companies whose continued existence depends on a process of production, distribution, and maintenance of the software. Information on much of the software can be obtained from GAMS, as we mentioned at the beginning of Chap. 12. Some of the free software can be obtained from `statlib` or `netlib`.

The following bibliography obviously covers a wide range of topics in statistical computing and computational statistics. Except for a few of the general references, all of these entries have been cited in the text.

© Springer International Publishing AG 2017

J.E. Gentle, *Matrix Algebra*, Springer Texts in Statistics,
DOI 10.1007/978-3-319-64867-5

Abramowitz, Milton, and Irene A. Stegun, eds. 1964. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. Washington: National Bureau of Standards (NIST). (Reprinted in 1965 by Dover Publications, Inc., New York.)

Alefeld, Göltz, and Jürgen Herzberger. (1983). *Introduction to Interval Computation*. New York: Academic Press.

Ammann, Larry, and John Van Ness. 1988. A routine for converting regression algorithms into corresponding orthogonal regression algorithms. *ACM Transactions on Mathematical Software* 14:76–87.

Anda, Andrew A., and Haesun Park. 1994. Fast plane rotations with dynamic scaling. *SIAM Journal of Matrix Analysis and Applications* 15:162–174.

Anda, Andrew A., and Haesun Park. 1996. Self-scaling fast rotations for stiff least squares problems. *Linear Algebra and Its Applications* 234:137–162.

Anderson, E., Z. Bai, C. Bischof, L. S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenhaum, S. Hammarling, A. McKenney, and D. Sorensen. 2000. *LAPACK Users' Guide*, 3rd ed. Philadelphia: Society for Industrial and Applied Mathematics.

Anderson, T. W. 1951. Estimating linear restrictions on regression coefficients for multivariate nomal distributions. *Annals of Mthematical Statistics* 22:327–351.

Anderson, T. W. 2003. *An Introduction to Multivariate Statistical Analysis*, 3rd ed. New York: John Wiley and Sons.

ANSI. 1978. *American National Standard for Information Systems — Programming Language FORTRAN*, Document X3.9-1978. New York: American National Standards Institute.

ANSI. 1989. *American National Standard for Information Systems — Programming Language C*, Document X3.159-1989. New York: American National Standards Institute.

ANSI. 1992. *American National Standard for Information Systems — Programming Language Fortran-90*, Document X3.9-1992. New York: American National Standards Institute.

ANSI. 1998. *American National Standard for Information Systems — Programming Language C++*, Document ISO/IEC 14882-1998. New York: American National Standards Institute.

Atkinson, A. C., and A. N. Donev. 1992. *Optimum Experimental Designs*. Oxford, United Kingdom: Oxford University Press.

Attaway, Stormy. 2016. *Matlab: A Practical Introduction to Programming and Problem Solving*, 4th ed. Oxford, United Kingdom: Butterworth-Heinemann.

Bailey, David H. 1993. Algorithm 719: Multiprecision translation and execution of FORTRAN programs. *ACM Transactions on Mathematical Software* 19:288–319.

Bailey, David H. 1995. A Fortran 90-based multiprecision system. *ACM Transactions on Mathematical Software* 21:379–387.

Bailey, David H., King Lee, and Horst D. Simon. 1990. Using Strassen's algorithm to accelerate the solution of linear systems. *Journal of Supercomputing* 4:358–371.

Bapat, R. B., and T. E. S. Raghavan. 1997. *Nonnegative Matrices and Applications*. Cambridge, United Kingdom: Cambridge University Press.

Barker, V. A., L. S. Blackford, J. Dongarra, J. Du Croz, S. Hammarling, M. Marinova, J. Wasniewsk, and P. Yalamov. 2001. *LAPACK95 Users' Guide*. Philadelphia: Society for Industrial and Applied Mathematics.

Barrett, R., M. Berry, T. F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. Van der Vorst. 1994. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*, 2nd ed. Philadelphia: Society for Industrial and Applied Mathematics.

Basilevsky, A. 1983. *Applied Matrix Algebra in the Statistical Sciences*. New York: North Holland

Beaton, Albert E., Donald B. Rubin, and John L. Barone. 1976. The acceptability of regression solutions: Another look at computational accuracy. *Journal of the American Statistical Association* 71:158–168.

Benzi, Michele. 2002. Preconditioning techniques for large linear systems: A survey. *Journal of Computational Physics* 182:418–477.

Bickel, Peter J., and Joseph A. Yahav. 1988. Richardson extrapolation and the bootstrap. *Journal of the American Statistical Association* 83:387–393.

Bindel, David, James Demmel, William Kahan, and Osni Marques. 2002. On computing Givens rotations reliably and efficiently. *ACM Transactions on Mathematical Software* 28:206–238.

Birkhoff, Garrett, and Surender Gulati. 1979. Isotropic distributions of test matrices. *Journal of Applied Mathematics and Physics (ZAMP)* 30:148–158.

Bischof, Christian H. 1990. Incremental condition estimation. *SIAM Journal of Matrix Analysis and Applications* 11:312–322.

Bischof, Christian H., and Gregorio Quintana-Ortí. 1998a. Computing rank-revealing QR factorizations. *ACM Transactions on Mathematical Software* 24:226–253.

Bischof, Christian H., and Gregorio Quintana-Ortí. 1998b. Algorithm 782: Codes for rank-revealing QR factorizations of dense matrices. *ACM Transactions on Mathematical Software* 24:254–257.

Björck, Åke. 1967. Solving least squares problems by Gram-Schmidt orthogonalization. *BIT* 7:1–21.

Björck, Åke. 1996. *Numerical Methods for Least Squares Problems*. Philadelphia: Society for Industrial and Applied Mathematics.

Blackford, L. S., J. Choi, A. Cleary, E. D'Azevedo, J. Demmel, I. Dhillon, J. Dongarra, S. Hammarling, G. Henry, A. Petitet, K. Stanley, D. Walker, and R. C. Whaley. 1997a. *ScaLAPACK Users' Guide*. Philadelphia: Society for Industrial and Applied Mathematics.

Blackford, L. S., A. Cleary, A. Petitet, R. C. Whaley, J. Demmel, I. Dhillon, H. Ren, K. Stanley, J. Dongarra, and S. Hammarling. 1997b. Practical ex-

perience in the numerical dangers of heterogeneous computing. *ACM Transactions on Mathematical Software* 23:133–147.

Blackford, L. Susan, Antoine Petitet, Roldan Pozo, Karin Remington, R. Clint Whaley, James Demmel, Jack Dongarra, Iain Duff, Sven Hammarling, Greg Henry, Michael Heroux, Linda Kaufman, and Andrew Lumsdaine. 2002. An updated set of basic linear algebra subprograms (BLAS). *ACM Transactions on Mathematical Software* 28:135–151.

Bollobás, Béla. 2013. *Modern Graph Theory*. New York: Springer-Verlag.

Brown, Peter N., and Homer F. Walker. 1997. GMRES on (nearly) singular systems. *SIAM Journal of Matrix Analysis and Applications* 18: 37–51.

Bunch, James R., and Linda Kaufman. 1977. Some stable methods for calculating inertia and solving symmetric linear systems. *Mathematics of Computation* 31:163–179.

Buttari, Alfredo, Julien Langou, Jakub Kurzak, and Jack Dongarra. 2009. A class of parallel tiled linear algebra algorithms for multicore architectures. *Parallel Computing* 35:38–53.

Calvetti, Daniela. 1991. Roundoff error for floating point representation of real data. *Communications in Statistics* 20:2687–2695.

Campbell, S. L., and C. D. Meyer, Jr. 1991. *Generalized Inverses of Linear Transformations*. New York: Dover Publications, Inc.

Carmeli, Moshe. 1983. *Statistical Theory and Random Matrices*. New York: Marcel Dekker, Inc.

Chaitin-Chatelin, Françoise, and Valérie Frayssé. 1996. *Lectures on Finite Precision Computations*. Philadelphia: Society for Industrial and Applied Mathematics.

Chambers, John M. 2016. *Extending R*. Boca Raton: Chapman and Hall/CRC Press.

Chan, T. F. 1982a. An improved algorithm for computing the singular value decomposition. *ACM Transactions on Mathematical Software* 8:72–83.

Chan, T. F. 1982b. Algorithm 581: An improved algorithm for computing the singular value decomposition. *ACM Transactions on Mathematical Software* 8:84–88.

Chan, T. F., G. H. Golub, and R. J. LeVeque. 1982. Updating formulae and a pairwise algorithm for computing sample variances. In *Compstat 1982: Proceedings in Computational Statistics*, ed. H. Caussinus, P. Ettinger, and R. Tomassone, 30–41. Vienna: Physica-Verlag.

Chan, Tony F., Gene H. Golub, and Randall J. LeVeque. 1983. Algorithms for computing the sample variance: Analysis and recommendations. *The American Statistician* 37:242–247.

Chapman, Barbara, Gabriele Jost, and Ruud van der Pas. 2007. *Using OpenMP: Portable Shared Memory Parallel Programming*. Cambridge, Massachusetts: The MIT Press.

Cheng, John, Max Grossman, and Ty McKercher. 2014. *Professional CUDA C Programming*. New York: Wrox Press, an imprint of John Wiley and Sons.

Chu, Moody T. 1991. Least squares approximation by real normal matrices with specified spectrum. *SIAM Journal on Matrix Analysis and Applications* 12:115–127.

Čížková, Lenka, and Pavel Čížek. 2012. Numerical linear algebra. In *Handbook of Computational Statistics: Concepts and Methods*, 2nd revised and updated ed., ed. James E. Gentle, Wolfgang Härdle, and Yuichi Mori, 105–137. Berlin: Springer.

Clerman, Norman, and Walter Spector. 2012. *Modern Fortran*. Cambridge, United Kingdom: Cambridge University Press.

Cline, Alan K., Andrew R. Conn, and Charles F. Van Loan. 1982. Generalizing the LINPACK condition estimator. In *Numerical Analysis, Mexico, 1981*, ed. J. P. Hennart, 73–83. Berlin: Springer-Verlag.

Cline, A. K., C. B. Moler, G. W. Stewart, and J. H. Wilkinson. 1979. An estimate for the condition number of a matrix. *SIAM Journal of Numerical Analysis* 16:368–375.

Cline, A. K., and R. K. Rew. 1983. A set of counter-examples to three condition number estimators. *SIAM Journal on Scientific and Statistical Computing* 4:602–611.

Cody, W. J. 1988. Algorithm 665: MACHAR: A subroutine to dynamically determine machine parameters. *ACM Transactions on Mathematical Software* 14:303–329.

Cody, W. J., and Jerome T. Coonen. 1993. Algorithm 722: Functions to support the IEEE standard for binary floating-point arithmetic. *ACM Transactions on Mathematical Software* 19:443–451.

Coleman, Thomas F., and Charles Van Loan. 1988. *Handbook for Matrix Computations*. Philadelphia: Society for Industrial and Applied Mathematics.

Cragg, John G., and Stephen G. Donald. 1996. On the asymptotic properties of LDU-based tests of the rank of a matrix. *Journal of the American Statistical Association* 91:1301–1309.

Cullen, M. R. 1985. *Linear Models in Biology*. New York: Halsted Press.

Dauger, Dean E., and Viktor K. Decyk. 2005. Plug-and-play cluster computing: High-performance computing for the mainstream. *Computing in Science and Engineering* 07(2):27–33.

Davies, Philip I., and Nicholas J. Higham. 2000. Numerically stable generation of correlation matrices and their factors. *BIT* 40:640–651.

Dempster, Arthur P., and Donald B. Rubin. 1983. Rounding error in regression: The appropriateness of Sheppard's corrections. *Journal of the Royal Statistical Society, Series B* 39:1–38.

Devlin, Susan J., R. Gnanadesikan, and J. R. Kettenring. 1975. Robust estimation and outlier detection with correlation coefficients. *Biometrika* 62:531–546.

Dey, Aloke, and Rahul Mukerjee. 1999. *Fractional Factorial Plans*. New York: John Wiley and Sons.

Dongarra, J. J., J. R. Bunch, C. B. Moler, and G. W. Stewart. 1979. *LINPACK Users' Guide*. Philadelphia: Society for Industrial and Applied Mathematics.

Dongarra, J. J., J. DuCroz, S. Hammarling, and I. Duff. 1990. A set of level 3 basic linear algebra subprograms. *ACM Transactions on Mathematical Software* 16:1–17.

Dongarra, J. J., J. DuCroz, S. Hammarling, and R. J. Hanson. 1988. An extended set of Fortran basic linear algebra subprograms. *ACM Transactions on Mathematical Software* 14:1–17.

Dongarra, Jack J., and Victor Eijkhout. 2000. Numerical linear algebra algorithms and software. *Journal of Computational and Applied Mathematics* 123:489–514.

Draper, Norman R., and Harry Smith. 1998. *Applied Regression Analysis*, 3rd ed. New York: John Wiley and Sons.

Duff, Iain S., Michael A. Heroux, and Roldan Pozo. 2002. An overview of the sparse basic linear algebra subprograms: the new standard from the BLAS technical forum. *ACM Transactions on Mathematical Software* 28:239–267.

Duff, Iain S., Michele Marrone, Guideppe Radicati, and Carlo Vittoli. 1997. Level 3 basic linear algebra subprograms for sparse matrices: A user-level interface. *ACM Transactions on Mathematical Software* 23:379–401.

Duff, Iain S., and Christof Vömel. 2002. Algorithm 818: A reference model implementation of the sparse BLAS in Fortran 95. *ACM Transactions on Mathematical Software* 28:268–283.

Eckart, Carl, and Gale Young. 1936. The approximation of one matrix by another of lower rank. *Psychometrika* 1:211–218.

Eddelbuettel, Dirk. 2013. *Seamless R and C++ Integration with Rcpp*. New York: Springer-Verlag.

Ericksen, Wilhelm S. 1985. Inverse pairs of test matrices. *ACM Transactions on Mathematical Software* 11:302–304.

Efron, Bradley, Trevor Hastie, Iain Johnstone, and Robert Tibshirani. 2004. Least angle regression. *The Annals of Statistics* 32:407–499.

Escobar, Luis A., and E. Barry Moser. 1993. A note on the updating of regression estimates. *The American Statistician* 47:192–194.

Eskow, Elizabeth, and Robert B. Schnabel. 1991. Algorithm 695: Software for a new modified Cholesky factorization. *ACM Transactions on Mathematical Software* 17:306–312.

Eubank, Randall L., and Ana Kupresanin. 2012. *Statistical Computing in C++ and R*. Boca Raton: Chapman and Hall/CRC Press.

Fasino, Dario, and Luca Gemignani. 2003. A Lanczos-type algorithm for the QR factorization of Cauchy-like matrices. In *Fast Algorithms for Structured Matrices: Theory and Applications*, ed. Vadim Olshevsky, 91–104. Providence, Rhode Island: American Mathematical Society.

Filippone, Salvatore, and Michele Colajanni. 2000. PSBLAS: A library for parallel linear algebra computation on sparse matrices. *ACM Transactions on Mathematical Software* 26:527–550.

Fuller, Wayne A. 1995. *Introduction to Statistical Time Series*, 2nd ed. New York: John Wiley and Sons.

Galassi, Mark, Jim Davies, James Theiler, Brian Gough, Gerard Jungman, Michael Booth, and Fabrice Rossi. 2002. *GNU Scientific Library Reference Manual*, 2nd ed. Bristol, United Kingdom: Network Theory Limited.

Gandrud, Christopher. 2015. *Reproducible Research with R and R Studio*, 2nd ed. Boca Raton: Chapman and Hall/CRC Press.

Gantmacher, F. R. 1959. *The Theory of Matrices*, Volumes I and II, translated by K. A. Hirsch, Chelsea, New York.

Geist, Al, Adam Beguelin, Jack Dongarra, Weicheng Jiang, Robert Manchek, and Vaidy Sunderam. 1994. *PVM. Parallel Virtual Machine. A Users' Guide and Tutorial for Networked Parallel Computing*. Cambridge, Massachusetts: The MIT Press.

Gentle, James E. 2003. *Random Number Generation and Monte Carlo Methods*, 2nd ed. New York: Springer-Verlag.

Gentle, James E. 2009. *Computational Statistics*. New York: Springer-Verlag.

Gentleman, W. M. 1974. Algorithm AS 75: Basic procedures for large, sparse or weighted linear least squares problems. *Applied Statistics* 23:448–454.

Gill, Len, and Arthur Lewbel. 1992. Testing the rank and definiteness of estimated matrices with applications to factor, state-space and ARMA models. *Journal of the American Statistical Association* 87:766–776.

Golub, G., and W. Kahan. 1965. Calculating the singular values and pseudo-inverse of a matrix. *SIAM Journal of Numerical Analysis, Series B* 2:205–224.

Golub, G. H., and C. Reinsch. 1970. Singular value decomposition and least squares solutions. *Numerische Mathematik* 14:403–420.

Golub, G. H., and C. F. Van Loan. 1980. An analysis of the total least squares problem. *SIAM Journal of Numerical Analysis* 17:883–893.

Golub, Gene H., and Charles F. Van Loan. 1996. *Matrix Computations*, 3rd ed. Baltimore: The Johns Hopkins Press.

Graybill, Franklin A. 1983. *Introduction to Matrices with Applications in Statistics*, 2nd ed. Belmont, California: Wadsworth Publishing Company.

Greenbaum, Anne, and Zdeněk Strakoš. 1992. Predicting the behavior of finite precision Lanczos and conjugate gradient computations. *SIAM Journal for Matrix Analysis and Applications* 13:121–137.

Gregory, Robert T., and David L. Karney. 1969. *A Collection of Matrices for Testing Computational Algorithms*. New York: John Wiley and Sons.

Gregory, R. T., and E. V. Krishnamurthy. 1984. *Methods and Applications of Error-Free Computation*. New York: Springer-Verlag.

Grewal, Mohinder S., and Angus P. Andrews. 1993. *Kalman Filtering Theory and Practice*. Englewood Cliffs, New Jersey: Prentice-Hall.

Griva, Igor, Stephen G. Nash, and Ariela Sofer. 2009. *Linear and Nonlinear Optimization*, 2nd ed. Philadelphia: Society for Industrial and Applied Mathematics.

Gropp, William D. 2005. Issues in accurate and reliable use of parallel computing in numerical programs. In *Accuracy and Reliability in Scientific Computing*, ed. Bo Einarsson, 253–263. Philadelphia: Society for Industrial and Applied Mathematics.

Gropp, William, Ewing Lusk, and Anthony Skjellum. 2014. *Using MPI: Portable Parallel Programming with the Message-Passing Interface*, 3rd ed. Cambridge, Massachusetts: The MIT Press.

Gropp, William, Ewing Lusk, and Thomas Sterling (Editors). 2003. *Beowulf Cluster Computing with Linux*, 2nd ed. Cambridge, Massachusetts: The MIT Press.

Haag, J. B., and D. S. Watkins. 1993. QR-like algorithms for the nonsymmetric eigenvalue problem. *ACM Transactions on Mathematical Software* 19:407–418.

Hager, W. W. 1984. Condition estimates. *SIAM Journal on Scientific and Statistical Computing* 5:311–316.

Hanson, Richard J., and Tim Hopkins. 2013. *Numerical Computing with Modern Fortran*. Philadelphia: Society for Industrial and Applied Mathematics.

Harville, David A. 1997. *Matrix Algebra from a Statistician's Point of View*. New York: Springer-Verlag.

Heath, M. T., E. Ng, and B. W. Peyton. 1991. Parallel algorithms for sparse linear systems. *SIAM Review* 33:420–460.

Hedayat, A. S., N. J. A. Sloane, and John Stufken. 1999. *Orthogonal Arrays: Theory and Applications*. New York: Springer-Verlag.

Heiberger, Richard M. 1978. Algorithm AS127: Generation of random orthogonal matrices. *Applied Statistics* 27:199–205.

Heroux, Michael A. 2015. Editorial: ACM TOMS replicated computational results initiative. *ACM Transactions on Mathematical Software* 41:Article No. 13.

Higham, Nicholas J. 1987. A survey of condition number estimation for triangular matrices. *SIAM Review* 29:575–596.

Higham, Nicholas J. 1988. FORTRAN codes for estimating the one-norm of a real or complex matrix, with applications to condition estimation. *ACM Transactions on Mathematical Software* 14:381–386.

Higham, Nicholas J. 1990. Experience with a matrix norm estimator. *SIAM Journal on Scientific and Statistical Computing* 11:804–809.

Higham, Nicholas J. 1991. Algorithm 694: A collection of test matrices in Matlab. *ACM Transactions on Mathematical Software* 17:289–305.

Higham, Nicholas J. 1997. Stability of the diagonal pivoting method with partial pivoting. *SIAM Journal of Matrix Analysis and Applications* 18:52–65.

Higham, Nicholas J. 2002. *Accuracy and Stability of Numerical Algorithms*, 2nd ed. Philadelphia: Society for Industrial and Applied Mathematics.

Higham, Nicholas J. 2008. *Functions of Matrices. Theory and Computation*. Philadelphia: Society for Industrial and Applied Mathematics.

Hill, Francis S., Jr., and Stephen M Kelley. 2006. *Computer Graphics Using OpenGL*, 3rd ed. New York: Pearson Education.

Hoffman, A. J., and H. W. Wielandt. 1953. The variation of the spectrum of a normal matrix. *Duke Mathematical Journal* 20:37–39.

Hong, H. P., and C. T. Pan. 1992. Rank-revealing QR factorization and SVD. *Mathematics of Computation* 58:213–232.

Horn, Roger A., and Charles R. Johnson. 1991. *Topics in Matrix Analysis*. Cambridge, United Kingdom: Cambridge University Press.

IEEE. 2008. *IEEE Standard for Floating-Point Arithmetic, Std 754-2008*. New York: IEEE, Inc.

Jansen, Paul, and Peter Weidner. 1986. High-accuracy arithmetic software — some tests of the ACRITH problem-solving routines. *ACM Transactions on Mathematical Software* 12:62–70.

Jaulin, Luc, Michel Kieffer, Olivier Didrit, and Eric Walter. (2001). *Applied Interval Analysis*. New York: Springer.

Jolliffe, I. T. 2002. *Principal Component Analysis*, 2nd ed. New York: Springer-Verlag.

Karau, Holden, Andy Konwinski, Patrick Wendell, and Matei Zaharia. 2015. *Learning Spark*. Sabastopol, California: O'Reilly Media, Inc.

Kearfott, R. Baker. 1996. INTERVAL_ARITHMETIC: A Fortran 90 module for an interval data type. *ACM Transactions on Mathematical Software* 22:385–392.

Kearfott, R. Baker, and Vladik Kreinovich (Editors). 1996. *Applications of Interval Computations*. Netherlands: Kluwer, Dordrecht.

Kearfott, R. B., M. Dawande, K. Du, and C. Hu. 1994. Algorithm 737: INTLIB: A portable Fortran 77 interval standard-function library. *ACM Transactions on Mathematical Software* 20:447–459.

Keller-McNulty, Sallie, and W. J. Kennedy. 1986. An error-free generalized matrix inversion and linear least squares method based on bordering. *Communications in Statistics — Simulation and Computation* 15:769–785.

Kennedy, William J., and James E. Gentle. 1980. *Statistical Computing*. New York: Marcel Dekker, Inc.

Kenney, C. S., and A. J. Laub. 1994. Small-sample statistical condition estimates for general matrix functions. *SIAM Journal on Scientific Computing* 15:191–209.

Kenney, C. S., A. J. Laub, and M. S. Reese. 1998. Statistical condition estimation for linear systems. *SIAM Journal on Scientific Computing* 19:566–583.

Kim, Hyunsoo, and Haesun Park. 2008. Nonnegative matrix factorization based on alternating non-negativity-constrained least squares and the active set method. *SIAM Journal on Matrix Analysis and Applications* 30:713–730.

Kleibergen, Frank, and Richard Paap. 2006. Generalized reduced rank tests using the singular value decomposition. *Journal of Econometrics* 133:97–126.

Kollo, Tõnu, and Dietrich von Rosen. 2005. *Advanced Multivariate Statistics with Matrices*. Amsterdam: Springer.

Kshemkalyani, Ajay D., and Mukesh Singhal. 2011. *Distributed Computing: Principles, Algorithms, and Systems*. Cambridge, United Kingdom: Cambridge University Press.

Kulisch, Ulrich. 2011. Very fast and exact accumulation of products. *Computing* 91:397–405.

Lawson, C. L., R. J. Hanson, D. R. Kincaid, and F. T. Krogh. 1979. Basic linear algebra subprograms for Fortran usage. *ACM Transactions on Mathematical Software* 5:308–323.

Lee, Daniel D., and H. Sebastian Seung. 2001. Algorithms for non-negative matrix factorization. *Advances in Neural Information Processing Systems*, 556–562. Cambridge, Massachusetts: The MIT Press.

Lemmon, David R., and Joseph L. Schafer. 2005. *Developing Statistical Software in Fortran 95*. New York: Springer-Verlag.

Leskovec, Jure, Anand Rajaraman, and Jeffrey David Ullman. 2014. *Mining of Massive Datasets*, 2nd ed. Cambridge, United Kingdom: Cambridge University Press.

Levesque, John, and Gene Wagenbreth. 2010. *High Performance Computing: Programming and Applications*. Boca Raton: Chapman and Hall/CRC Press.

Liem, C. B., T. Lü, and T. M. Shih. 1995. *The Splitting Extrapolation Method*. Singapore: World Scientific.

Linnainmaa, Seppo. 1975. Towards accurate statistical estimation of rounding errors in floating-point computations. *BIT* 15:165–173.

Liu, Shuangzhe and Heinz Neudecker. 1996. Several matrix Kantorovich-type inequalities. *Journal of Mathematical Analysis and Applications* 197:23–26.

Loader, Catherine. 2012. Smoothing: Local regression techniques. In *Handbook of Computational Statistics: Concepts and Methods*, 2nd revised and updated ed., ed. James E. Gentle, Wolfgang Härdle, and Yuichi Mori, 571–596. Berlin: Springer.

Longley, James W. 1967. An appraisal of least squares problems for the electronic computer from the point of view of the user. *Journal of the American Statistical Association* 62:819–841.

Luk, F. T., and H. Park. 1989. On parallel Jacobi orderings. *SIAM Journal on Scientific and Statistical Computing* 10:18–26.

Magnus, Jan R., and Heinz Neudecker. 1999. *Matrix Differential Calculus with Applications in Statistics and Econometrics*, revised ed. New York: John Wiley and Sons.

Markus, Arjen. 2012. *Modern Fortran in Practice*. Cambridge, United Kingdom: Cambridge University Press.

Marshall, A. W., and I. Olkin. 1990. Matrix versions of the Cauchy and Kantorovich inequalities. *Aequationes Mathematicae* 40:89–93.

Metcalf, Michael, John Reid, and Malcolm Cohen. 2011. *Modern Fortran Explained*. Oxford, United Kingdom: Oxford University Press.

Meyn, Sean, and Richard L. Tweedie. 2009. *Markov Chains and Stochastic Stability*, 2nd ed. Cambridge, United Kingdom: Cambridge University Press.

Miller, Alan J. 1992. Algorithm AS 274: Least squares routines to supplement those of Gentleman. *Applied Statistics* 41:458–478 (Corrections, 1994, *ibid.* 43:678).

Miller, Alan. 2002. *Subset Selection in Regression*, 2nd ed. Boca Raton: Chapman and Hall/CRC Press.

Miller, Alan J., and Nam-Ky Nguyen. 1994. A Fedorov exchange algorithm for D-optimal design. *Applied Statistics* 43:669–678.

Mizuta, Masahiro. 2012. Dimension reduction methods. In *Handbook of Computational Statistics: Concepts and Methods*, 2nd revised and updated ed., ed. James E. Gentle, Wolfgang Härdle, and Yuichi Mori, 619–644. Berlin: Springer.

Moore, E. H. 1920. On the reciprocal of the general algebraic matrix. *Bulletin of the American Mathematical Society* 26:394–395.

Moore, Ramon E. (1979). *Methods and Applications of Interval Analysis*. Philadelphia: Society for Industrial and Applied Mathematics.

Mosteller, Frederick, and David L. Wallace. 1963. Inference in an authorship problem. *Journal of the American Statistical Association* 58:275–309.

Muirhead, Robb J. 1982. *Aspects of Multivariate Statistical Theory*. New York: John Wiley and Sons.

Mullet, Gary M., and Tracy W. Murray. 1971. A new method for examining rounding error in least-squares regression computer programs. *Journal of the American Statistical Association* 66:496–498.

Nachbin, Leopoldo. 1965. *The Haar Integral*, translated by Lulu Bechtolsheim. Princeton, New Jersey: D. Van Nostrand Co Inc.

Nakano, Junji. 2012. Parallel computing techniques. In *Handbook of Computational Statistics: Concepts and Methods*, 2nd revised and updated ed., ed. James E. Gentle, Wolfgang Härdle, and Yuichi Mori, 243–272. Berlin: Springer.

Nguyen, Nam-Ky, and Alan J. Miller. 1992. A review of some exchange algorithms for constructing D-optimal designs. *Computational Statistics and Data Analysis* 14:489–498.

Olshevsky, Vadim (Editor). 2003. *Fast Algorithms for Structured Matrices: Theory and Applications*. Providence, Rhode Island: American Mathematical Society.

Olver, Frank W. J., Daniel w. Lozier, Ronald F. Boisvert, and Charles W. Clark. 2010. *NIST Handbook of Mathematical Functions*. Cambridge: Cambridge University Press.

Overton, Michael L. 2001. *Numerical Computing with IEEE Floating Point Arithmetic*. Philadelphia: Society for Industrial and Applied Mathematics.

Parsian, Mahmoud. 2015. *Data Algorithms*. Sabastopol, California: O'Reilly Media, Inc.

Penrose, R. 1955. A generalized inverse for matrices. *Proceedings of the Cambridge Philosophical Society* 51:406–413.

Quinn, Michael J. 2003. *Parallel Programming in C with MPI and OpenMP*. New York: McGraw-Hill.

Rice, John R. 1966. Experiments on Gram-Schmidt orthogonalization. *Mathematics of Computation* 20:325–328.

Rice, John R. 1993. *Numerical Methods, Software, and Analysis*, 2nd ed. New York: McGraw-Hill Book Company.

Robin, J. M., and R. J. Smith. 2000. Tests of rank. *Econometric Theory* 16:151–175.

Roosta, Seyed H. 2000. *Parallel Processing and Parallel Algorithms: Theory and Computation*. New York: Springer-Verlag.

Rousseeuw, Peter J., and Geert Molenberghs. 1993. Transformation of nonpositive semidefinite correlation matrices. *Communications in Statistics — Theory and Methods* 22:965–984.

Rust, Bert W. 1994. Perturbation bounds for linear regression problems. *Computing Science and Statistics* 26:528–532.

Saad, Y., and M. H. Schultz. 1986. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing* 7:856–869.

Schott, James R. 2004. *Matrix Analysis for Statistics*, 2nd ed. New York: John Wiley and Sons.

Searle, S. R. 1971. *Linear Models*. New York: John Wiley and Sons.

Searle, Shayle R. 1982. *Matrix Algebra Useful for Statistics*. New York: John Wiley and Sons.

Shao, Jun. 2003. *Mathematical Statistics*, 2nd ed. New York: Springer-Verlag.

Sherman, J., and W. J. Morrison. 1950. Adjustment of an inverse matrix corresponding to a change in one element of a given matrix. *Annals of Mathematical Statistics* 21:124–127.

Siek, Jeremy, and Andrew Lumsdaine. 2000. A modern framework for portable high-performance numerical linear algebra. In *Advances in Software Tools for Scientific Computing*, ed. Are Bruaset, H. Langtangen, and E. Quak, 1–56. New York: Springer-Verlag.

Skeel, R. D. 1980. Iterative refinement implies numerical stability for Gaussian elimination. *Mathematics of Computation* 35:817–832.

Smith, B. T., J. M. Boyle, J. J. Dongarra, B. S. Garbow, Y. Ikebe, V. C. Klema, and C. B. Moler. 1976. *Matrix Eigensystem Routines — EISPACK Guide*. Berlin: Springer-Verlag.

Stallings, W. T., and T. L. Boullion. 1972. Computation of pseudo-inverse using residue arithmetic. *SIAM Review* 14:152–163.

Stewart, G. W. 1980. The efficient generation of random orthogonal matrices with an application to condition estimators. *SIAM Journal of Numerical Analysis* 17:403–409.

Stewart, G. W. 1990. Stochastic perturbation theory. *SIAM Review* 32:579–610.

Stodden, Victoria, Friedrich Leisch, and Roger D. Peng. 2014. *Implementing Reproducible Research*. Boca Raton: Chapman and Hall/CRC Press.

Strang, Gilbert, and Tri Nguyen. 2004. The interplay of ranks of submatrices. *SIAM Review* 46:637–646.

Strassen, V. 1969, Gaussian elimination is not optimal. *Numerische Mathematik* 13:354–356.

Szabó, S., and R. Tanaka. 1967. *Residue Arithmetic and Its Application to Computer Technology*. New York: McGraw-Hill.

Tanner, M. A., and R. A. Thisted. 1982. A remark on AS127. Generation of random orthogonal matrices. *Applied Statistics* 31:190–192.

Titterington, D. M. 1975. Optimal design: Some geometrical aspects of *D*-optimality. *Biometrika* 62:313–320.

Trefethen, Lloyd N., and Mark Embree. 2005. *Spectra and Pseudospectra: The Behavior of Nonnormal Matrices and Operators*. Princeton: Princeton University Press.

Trefethen, Lloyd N., and David Bau III. 1997. *Numerical Linear Algebra*. Philadelphia: Society for Industrial and Applied Mathematics.

Trosset, Michael W. 2002. Extensions of classical multidimensional scaling via variable reduction. *Computational Statistics* 17:147–163.

Unicode Consortium. 1990. *The Unicode Standard, Worldwide Character Encoding, Version 1.0, Volume 1*. Reading, Massachusetts: Addison-Wesley Publishing Company.

Unicode Consortium. 1992. *The Unicode Standard, Worldwide Character Encoding, Version 1.0, Volume 2*. Reading, Massachusetts: Addison-Wesley Publishing Company.

Vandenberghe, Lieven, and Stephen Boyd. 1996. Semidefinite programming. *SIAM Review* 38:49–95.

Venables, W. N., and B. D. Ripley. 2003. *Modern Applied Statistics with S*, 4th ed. New York: Springer-Verlag.

Walker, Homer F. 1988. Implementation of the GMRES method using Householder transformations. *SIAM Journal on Scientific and Statistical Computing* 9:152–163.

Walker, Homer F., and Lu Zhou. 1994. A simpler GMRES. *Numerical Linear Algebra with Applications* 1:571–581.

Walster, G. William. 1996. Stimulating hardware and software support for interval arithmetic. In *Applications of Interval Computations*, ed. R. Baker Kearfott and Vladik Kreinovich, 405–416. Dordrecht, Netherlands: Kluwer.

Walster, G. William. 2005. The use and implementation of interval data types. In *Accuracy and Reliability in Scientific Computing*, ed. Bo Einarsson, 173–194. Philadelphia: Society for Industrial and Applied Mathematics.

Watkins, David S. 2002. *Fundamentals of Matrix Computations*, 2nd ed. New York: John Wiley and Sons.

White, Tom. 2015. *Hadoop: The Definitive Guide*, 4th ed. Sabastopol, California: O'Reilly Media, Inc.

Wickham, Hadley. 2015) *Advanced R*. Boca Raton: Chapman and Hall/CRC Press.

Wilkinson, J. H. 1959. The evaluation of the zeros of ill-conditioned polynomials. *Numerische Mathematik* 1:150–180.

Wilkinson, J. H. 1963. *Rounding Errors in Algebraic Processes*. Englewood Cliffs, New Jersey: Prentice-Hall. (Reprinted by Dover Publications, Inc., New York, 1994).

Wilkinson, J. H. 1965. *The Algebraic Eigenvalue Problem*. New York: Oxford University Press.

Woodbury, M. A. 1950. "Inverting Modified Matrices", Memorandum Report 42, Statistical Research Group, Princeton University.

Wynn, P. 1962. Acceleration techniques for iterated vector and matrix problems. *Mathematics of Computation* 16:301–322.

Xie, Yihui. 2015. *Dynamic Documents with R and knitr*, 2nd ed. Boca Raton: Chapman and Hall/CRC Press.

Zhou, Bing Bing, and Richard P. Brent. 2003. An efficient method for computing eigenvalues of a real normal matrix. *Journal of Parallel and Distributed Computing* 63:638–648.

Index

© Springer International Publishing AG 2017

J.E. Gentle, *Matrix Algebra*, Springer Texts in Statistics,
DOI 10.1007/978-3-319-64867-5

T