Introduction to Applied Linear Algebra

Vectors, Matrices, and Least Squares

Stephen Boyd

Department of Electrical Engineering
Stanford University

Lieven Vandenberghe

Department of Electrical and Computer Engineering
University of California, Los Angeles

CAMBRIDGE
UNIVERSITY PRESS

CAMBRIDGE
UNIVERSITY PRESS

University Printing House, Cambridge CB2 8BS, United Kingdom

One Liberty Plaza, 20th Floor, New York, NY 10006, USA

477 Williamstown Road, Port Melbourne, VIC 3207, Australia

314–321, 3rd Floor, Plot 3, Splendor Forum, Jasola District Centre,
New Delhi – 110025, India

79 Anson Road, #06–04/06, Singapore 079906

Cambridge University Press is part of the University of Cambridge.

It furthers the University's mission by disseminating knowledge in the pursuit of education, learning, and research at the highest international levels of excellence.

www.cambridge.org
Information on this title: www.cambridge.org/9781316518960
DOI: 10.1017/9781108583664

First published 2018

Printed in the United Kingdom by Clays, St Ives plc, 2018

A catalogue record for this publication is available from the British Library.

ISBN 978-1-316-51896-0 Hardback

Additional resources for this publication at www.cambridge.org/IntroAppLinAlg

For

Anna, Nicholas, and Nora

Daniël and Margriet

Contents

Preface

This book is meant to provide an introduction to vectors, matrices, and least squares methods, basic topics in applied linear algebra. Our goal is to give the beginning student, with little or no prior exposure to linear algebra, a good grounding in the basic ideas, as well as an appreciation for how they are used in many applications, including data fitting, machine learning and artificial intelligence, tomography, navigation, image processing, finance, and automatic control systems.

The background required of the reader is familiarity with basic mathematical notation. We use calculus in just a few places, but it does not play a critical role and is not a strict prerequisite. Even though the book covers many topics that are traditionally taught as part of probability and statistics, such as fitting mathematical models to data, no knowledge of or background in probability and statistics is needed.

The book covers less mathematics than a typical text on applied linear algebra. We use only one theoretical concept from linear algebra, linear independence, and only one computational tool, the QR factorization; our approach to most applications relies on only one method, least squares (or some extension). In this sense we aim for intellectual economy: With just a few basic mathematical ideas, concepts, and methods, we cover many applications. The mathematics we do present, however, is complete, in that we carefully justify every mathematical statement. In contrast to most introductory linear algebra texts, however, we describe many applications, including some that are typically considered advanced topics, like document classification, control, state estimation, and portfolio optimization.

The book does not require any knowledge of computer programming, and can be used as a conventional textbook, by reading the chapters and working the exercises that do not involve numerical computation. This approach however misses out on one of the most compelling reasons to learn the material: You can use the ideas and methods described in this book to do practical things like build a prediction model from data, enhance images, or optimize an investment portfolio. The growing power of computers, together with the development of high level computer languages and packages that support vector and matrix computation, have made it easy to use the methods described in this book for real applications. For this reason we hope that every student of this book will complement their study with computer programming exercises and projects, including some that involve real data. This book includes some generic exercises that require computation; additional ones, and the associated data files and language-specific resources, are available online.

If you read the whole book, work some of the exercises, and carry out computer exercises to implement or use the ideas and methods, you will learn a lot. While there will still be much for you to learn, you will have seen many of the basic ideas behind modern data science and other application areas. We hope you will be empowered to use the methods for your own applications.

The book is divided into three parts. Part I introduces the reader to vectors, and various vector operations and functions like addition, inner product, distance, and angle. We also describe how vectors are used in applications to represent word counts in a document, time series, attributes of a patient, sales of a product, an audio track, an image, or a portfolio of investments. Part II does the same for matrices, culminating with matrix inverses and methods for solving linear equations. Part III, on least squares, is the payoff, at least in terms of the applications. We show how the simple and natural idea of approximately solving a set of overdetermined equations, and a few extensions of this basic idea, can be used to solve many practical problems.

The whole book can be covered in a 15 week (semester) course; a 10 week (quarter) course can cover most of the material, by skipping a few applications and perhaps the last two chapters on nonlinear least squares. The book can also be used for self-study, complemented with material available online. By design, the pace of the book accelerates a bit, with many details and simple examples in parts I and II, and more advanced examples and applications in part III. A course for students with little or no background in linear algebra can focus on parts I and II, and cover just a few of the more advanced applications in part III. A more advanced course on applied linear algebra can quickly cover parts I and II as review, and then focus on the applications in part III, as well as additional topics.

We are grateful to many of our colleagues, teaching assistants, and students for helpful suggestions and discussions during the development of this book and the associated courses. We especially thank our colleagues Trevor Hastie, Rob Tibshirani, and Sanjay Lall, as well as Nick Boyd, for discussions about data fitting and classification, and Jenny Hong, Ahmed Bou-Rabee, Keegan Go, David Zeng, and Jaehyun Park, Stanford undergraduates who helped create and teach the course EE103. We thank David Tse, Alex Lemon, Neal Parikh, and Julie Lancashire for carefully reading drafts of this book and making many good suggestions.

Stephen Boyd — *Stanford, California*
Lieven Vandenberghe — *Los Angeles, California*

Part I

Vectors

Chapter 1

Vectors

In this chapter we introduce vectors and some common operations on them. We describe some settings in which vectors are used.

1.1 Vectors

A *vector* is an ordered finite list of numbers. Vectors are usually written as vertical arrays, surrounded by square or curved brackets, as in

$$\begin{bmatrix} -1.1 \\ 0.0 \\ 3.6 \\ -7.2 \end{bmatrix} \quad \text{or} \quad \begin{pmatrix} -1.1 \\ 0.0 \\ 3.6 \\ -7.2 \end{pmatrix}.$$

They can also be written as numbers separated by commas and surrounded by parentheses. In this notation style, the vector above is written as

$$(-1.1, 0.0, 3.6, -7.2).$$

The *elements* (or *entries*, *coefficients*, *components*) of a vector are the values in the array. The *size* (also called *dimension* or *length*) of the vector is the number of elements it contains. The vector above, for example, has size four; its third entry is 3.6. A vector of size n is called an *n-vector*. A 1-vector is considered to be the same as a number, *i.e.*, we do not distinguish between the 1-vector $[\ 1.3\ ]$ and the number 1.3.

We often use symbols to denote vectors. If we denote an n-vector using the symbol a, the ith element of the vector a is denoted a_i, where the subscript i is an integer index that runs from 1 to n, the size of the vector.

Two vectors a and b are *equal*, which we denote $a = b$, if they have the same size, and each of the corresponding entries is the same. If a and b are n-vectors, then $a = b$ means $a_1 = b_1, \ldots, a_n = b_n$.

The numbers or values of the elements in a vector are called *scalars*. We will focus on the case that arises in most applications, where the scalars are real numbers. In this case we refer to vectors as *real vectors*. (Occasionally other types of scalars arise, for example, complex numbers, in which case we refer to the vector as a *complex vector*.) The set of all real numbers is written as $\mathbf{R}$, and the set of all real n-vectors is denoted $\mathbf{R}^n$, so $a \in \mathbf{R}^n$ is another way to say that a is an n-vector with real entries. Here we use set notation: $a \in \mathbf{R}^n$ means that a is an element of the set $\mathbf{R}^n$; see appendix A.

Block or stacked vectors. It is sometimes useful to define vectors by *concatenating* or *stacking* two or more vectors, as in

$$a = \begin{bmatrix} b \\ c \\ d \end{bmatrix},$$

where a, b, c, and d are vectors. If b is an m-vector, c is an n-vector, and d is a p-vector, this defines the $(m+n+p)$-vector

$$a = (b_1, b_2, \ldots, b_m, c_1, c_2, \ldots, c_n, d_1, d_2, \ldots, d_p).$$

The stacked vector a is also written as $a = (b, c, d)$.

Stacked vectors can include scalars (numbers). For example if a is a 3-vector, $(1, a)$ is the 4-vector $(1, a_1, a_2, a_3)$.

Subvectors. In the equation above, we say that b, c, and d are *subvectors* or *slices* of a, with sizes m, n, and p, respectively. *Colon notation* is used to denote subvectors. If a is a vector, then $a_{r:s}$ is the vector of size $s - r + 1$, with entries $a_r, \ldots, a_s$:

$$a_{r:s} = (a_r, \ldots, a_s).$$

The subscript $r:s$ is called the *index range*. Thus, in our example above, we have

$$b = a_{1:m}, \qquad c = a_{(m+1):(m+n)}, \qquad d = a_{(m+n+1):(m+n+p)}.$$

As a more concrete example, if z is the 4-vector $(1, -1, 2, 0)$, the slice $z_{2:3}$ is $z_{2:3} = (-1, 2)$. Colon notation is not completely standard, but it is growing in popularity.

Notational conventions. Some authors try to use notation that helps the reader distinguish between vectors and scalars (numbers). For example, Greek letters (α, β, ...) might be used for numbers, and lower-case letters (a, x, f, ...) for vectors. Other notational conventions include vectors given in bold font ($\mathbf{g}$), or vectors written with arrows above them ($\vec{a}$). These notational conventions are not standardized, so you should be prepared to figure out what things are (*i.e.*, scalars or vectors) despite the author's notational scheme (if any exists).

Indexing. We should give a couple of warnings concerning the subscripted index notation a_i. The first warning concerns the range of the index. In many computer languages, arrays of length n are indexed from $i = 0$ to $i = n - 1$. But in standard mathematical notation, n-vectors are indexed from $i = 1$ to $i = n$, so in this book, vectors will be indexed from $i = 1$ to $i = n$.

The next warning concerns an ambiguity in the notation a_i, used for the ith element of a vector a. The same notation will occasionally refer to the ith vector in a collection or list of k vectors $a_1, \ldots, a_k$. Whether a_3 means the third element of a vector a (in which case a_3 is a number), or the third vector in some list of vectors (in which case a_3 is a vector) should be clear from the context. When we need to refer to an element of a vector that is in an indexed collection of vectors, we can write $(a_i)_j$ to refer to the jth entry of a_i, the ith vector in our list.

Zero vectors. A *zero vector* is a vector with all elements equal to zero. Sometimes the zero vector of size n is written as 0_n, where the subscript denotes the size. But usually a zero vector is denoted just 0, the same symbol used to denote the number 0. In this case you have to figure out the size of the zero vector from the context. As a simple example, if a is a 9-vector, and we are told that $a = 0$, the 0 vector on the right-hand side must be the one of size 9.

Even though zero vectors of different sizes are different vectors, we use the same symbol 0 to denote them. In computer programming this is called *overloading*: The symbol 0 is overloaded because it can mean different things depending on the context (*e.g.*, the equation it appears in).

Unit vectors. A (standard) *unit vector* is a vector with all elements equal to zero, except one element which is equal to one. The ith unit vector (of size n) is the unit vector with ith element one, and denoted e_i. For example, the vectors

$$e_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \qquad e_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \qquad e_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

are the three unit vectors of size 3. The notation for unit vectors is an example of the ambiguity in notation noted above. Here, e_i denotes the ith unit vector, and not the ith element of a vector e. Thus we can describe the ith unit n-vector e_i as

$$(e_i)_j = \begin{cases} 1 & j = i \\ 0 & j \neq i, \end{cases}$$

for $i, j = 1, \ldots, n$. On the left-hand side e_i is an n-vector; $(e_i)_j$ is a number, its jth entry. As with zero vectors, the size of e_i is usually determined from the context.

Ones vector. We use the notation $\mathbf{1}_n$ for the n-vector with all its elements equal to one. We also write $\mathbf{1}$ if the size of the vector can be determined from the context. (Some authors use e to denote a vector of all ones, but we will not use this notation.) The vector $\mathbf{1}$ is sometimes called the *ones vector*.

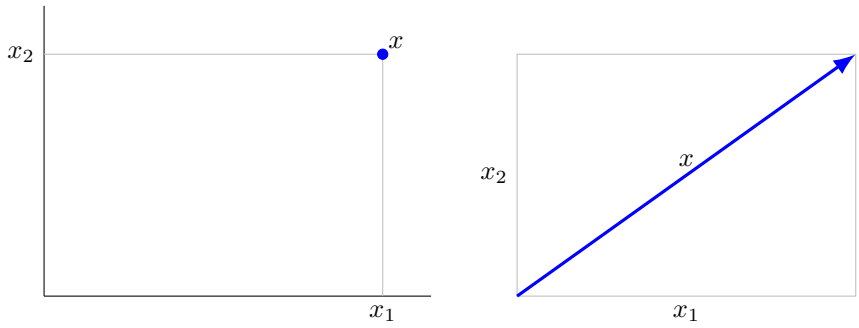

Figure 1.1 *Left.* The 2-vector x specifies the position (shown as a dot) with coordinates x_1 and x_2 in a plane. *Right.* The 2-vector x represents a displacement in the plane (shown as an arrow) by x_1 in the first axis and x_2 in the second.

Sparsity. A vector is said to be *sparse* if many of its entries are zero; its *sparsity pattern* is the set of indices of nonzero entries. The number of the nonzero entries of an n-vector x is denoted $\mathbf{nnz}(x)$. Unit vectors are sparse, since they have only one nonzero entry. The zero vector is the sparsest possible vector, since it has no nonzero entries. Sparse vectors arise in many applications.

Examples

An n-vector can be used to represent n quantities or values in an application. In some cases the values are similar in nature (for example, they are given in the same physical units); in others, the quantities represented by the entries of the vector are quite different from each other. We briefly describe below some typical examples, many of which we will see throughout the book.

Location and displacement. A 2-vector can be used to represent a position or location in a 2-dimensional (2-D) space, *i.e.*, a plane, as shown in figure 1.1. A 3-vector is used to represent a location or position of some point in 3-dimensional (3-D) space. The entries of the vector give the coordinates of the position or location.

A vector can also be used to represent a displacement in a plane or 3-D space, in which case it is typically drawn as an arrow, as shown in figure 1.1. A vector can also be used to represent the velocity or acceleration, at a given time, of a point that moves in a plane or 3-D space.

Color. A 3-vector can represent a color, with its entries giving the Red, Green, and Blue (RGB) intensity values (often between 0 and 1). The vector $(0, 0, 0)$ represents black, the vector $(0, 1, 0)$ represents a bright pure green color, and the vector $(1, 0.5, 0.5)$ represents a shade of pink. This is illustrated in figure 1.2.

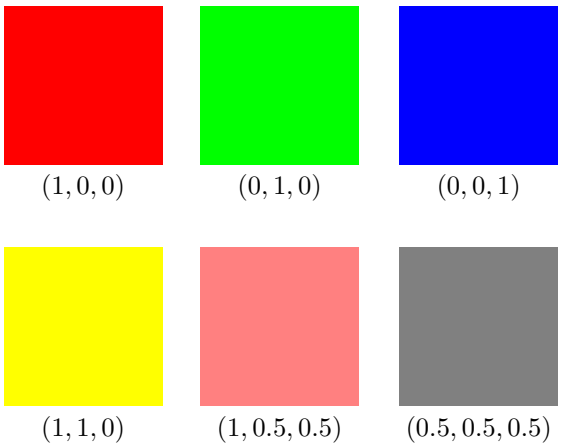

Figure 1.2 Six colors and their RGB vectors.

Quantities. An n-vector q can represent the amounts or quantities of n different resources or products held (or produced, or required) by an entity such as a company. Negative entries mean an amount of the resource owed to another party (or consumed, or to be disposed of). For example, a *bill of materials* is a vector that gives the amounts of n resources required to create a product or carry out a task.

Portfolio. An n-vector s can represent a stock portfolio or investment in n different assets, with s_i giving the number of shares of asset i held. The vector $(100, 50, 20)$ represents a portfolio consisting of 100 shares of asset 1, 50 shares of asset 2, and 20 shares of asset 3. Short positions (*i.e.*, shares that you owe another party) are represented by negative entries in a portfolio vector. The entries of the portfolio vector can also be given in dollar values, or fractions of the total dollar amount invested.

Values across a population. An n-vector can give the values of some quantity across a population of individuals or entities. For example, an n-vector b can give the blood pressure of a collection of n patients, with b_i the blood pressure of patient i, for $i = 1, \ldots, n$.

Proportions. A vector w can be used to give fractions or proportions out of n choices, outcomes, or options, with w_i the fraction with choice or outcome i. In this case the entries are nonnegative and add up to one. Such vectors can also be interpreted as the recipes for a mixture of n items, an allocation across n entities, or as probability values in a probability space with n outcomes. For example, a uniform mixture of 4 outcomes is represented as the 4-vector $(1/4, 1/4, 1/4, 1/4)$.

Time series. An n-vector can represent a *time series* or *signal*, that is, the value of some quantity at different times. (The entries in a vector that represents a time series are sometimes called *samples*, especially when the quantity is something

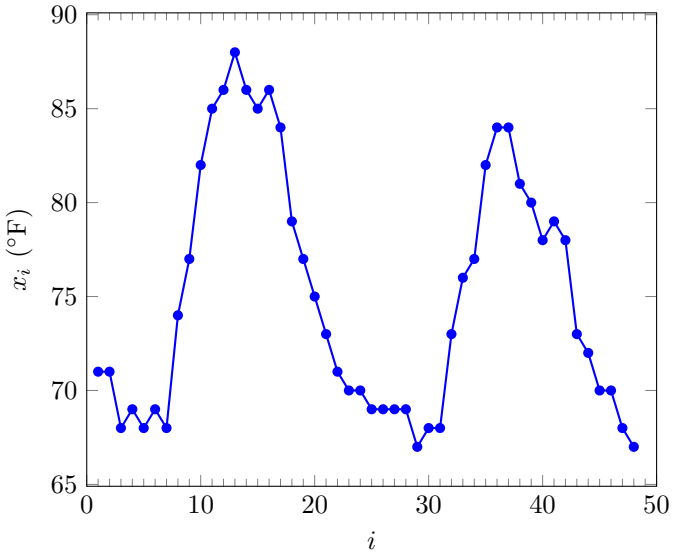

Figure 1.3 Hourly temperature in downtown Los Angeles on August 5 and 6, 2015 (starting at 12:47AM, ending at 11:47PM).

measured.) An audio (sound) signal can be represented as a vector whose entries give the value of acoustic pressure at equally spaced times (typically 48000 or 44100 per second). A vector might give the hourly rainfall (or temperature, or barometric pressure) at some location, over some time period. When a vector represents a time series, it is natural to plot x_i versus i with lines connecting consecutive time series values. (These lines carry no information; they are added only to make the plot easier to understand visually.) An example is shown in figure 1.3, where the 48-vector x gives the hourly temperature in downtown Los Angeles over two days.

Daily return. A vector can represent the daily return of a stock, *i.e.*, its fractional increase (or decrease if negative) in value over the day. For example the return time series vector $(-0.022, +0.014, +0.004)$ means the stock price went down 2.2% on the first day, then up 1.4% the next day, and up again 0.4% on the third day. In this example, the samples are not uniformly spaced in time; the index refers to trading days, and does not include weekends or market holidays. A vector can represent the daily (or quarterly, hourly, or minute-by-minute) value of any other quantity of interest for an asset, such as price or volume.

Cash flow. A cash flow into and out of an entity (say, a company) can be represented by a vector, with positive entries representing payments to the entity, and negative entries representing payments by the entity. For example, with entries giving cash flow each quarter, the vector $(1000, -10, -10, -10, -1010)$ represents a one year loan of \$1000, with 1% interest only payments made each quarter, and the principal and last interest payment at the end.

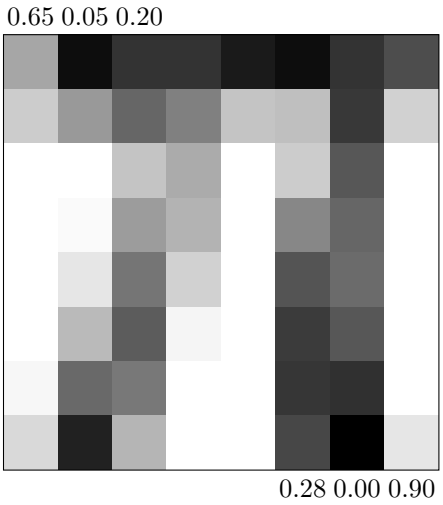

Figure 1.4 8×8 image and the grayscale levels at six pixels.

Images. A monochrome (black and white) image is an array of $M \times N$ pixels (square patches with uniform grayscale level) with M rows and N columns. Each of the MN pixels has a grayscale or intensity value, with 0 corresponding to black and 1 corresponding to bright white. (Other ranges are also used.) An image can be represented by a vector of length MN, with the elements giving grayscale levels at the pixel locations, typically ordered column-wise or row-wise.

Figure 1.4 shows a simple example, an 8×8 image. (This is a very low resolution; typical values of M and N are in the hundreds or thousands.) With the vector entries arranged row-wise, the associated 64-vector is

$$x = (0.65, 0.05, 0.20, \ldots, 0.28, 0.00, 0.90).$$

A color $M \times N$ pixel image is described by a vector of length $3MN$, with the entries giving the R, G, and B values for each pixel, in some agreed-upon order.

Video. A monochrome video, *i.e.*, a sequence of length K of images with $M \times N$ pixels, can be represented by a vector of length KMN (again, in some particular order).

Word count and histogram. A vector of length n can represent the number of times each word in a dictionary of n words appears in a document. For example, $(25, 2, 0)$ means that the first dictionary word appears 25 times, the second one twice, and the third one not at all. (Typical dictionaries used for document word counts have many more than 3 elements.) A small example is shown in figure 1.5. A variation is to have the entries of the vector give the *histogram* of word frequencies in the document, so that, *e.g.*, $x_5 = 0.003$ means that 0.3% of all the words in the document are the fifth word in the dictionary.

It is common practice to count variations of a word (say, the same word stem with different endings) as the same word; for example, 'rain', 'rains', 'raining', and

Word count vectors are used in computer based document analysis. Each entry of the word count vector is the number of times the associated dictionary word appears in the document.

$$\begin{array}{ll} \text{word} & \left[\begin{array}{c} 3 \\ 2 \\ 1 \\ 0 \\ 4 \\ 2 \end{array}\right] \\ \text{in} & \\ \text{number} & \\ \text{horse} & \\ \text{the} & \\ \text{document} & \end{array}$$

Figure 1.5 A snippet of text (top), the dictionary (bottom left), and word count vector (bottom right).

'rained' might all be counted as 'rain'. Reducing each word to its stem is called *stemming*. It is also common practice to exclude words that are too common (such as 'a' or 'the'), which are referred to as *stop words*, as well as words that are extremely rare.

Customer purchases. An n-vector p can be used to record a particular customer's purchases from some business over some period of time, with p_i the quantity of item i the customer has purchased, for $i = 1, \ldots, n$. (Unless n is small, we would expect many of these entries to be zero, meaning the customer has not purchased those items.) In one variation, p_i represents the total dollar value of item i the customer has purchased.

Occurrence or subsets. An n-vector o can be used to record whether or not each of n different events has occurred, with $o_i = 0$ meaning that event i did not occur, and $o_i = 1$ meaning that it did occur. Such a vector encodes a subset of a collection of n objects, with $o_i = 1$ meaning that object i is contained in the subset, and $o_i = 0$ meaning that object i is not in the subset. Each entry of the vector a is either 0 or 1; such vectors are called *Boolean*, after the mathematician George Boole, a pioneer in the study of logic.

Features or attributes. In many applications a vector collects together n different quantities that pertain to a single thing or object. The quantities can be measurements, or quantities that can be measured or derived from the object. Such a vector is sometimes called a *feature vector*, and its entries are called the *features* or *attributes*. For example, a 6-vector f could give the age, height, weight, blood pressure, temperature, and gender of a patient admitted to a hospital. (The last entry of the vector could be encoded as $f_6 = 0$ for male, $f_6 = 1$ for female.) In this example, the quantities represented by the entries of the vector are quite different, with different physical units.

Vector entry labels. In applications such as the ones described above, each entry of a vector has a meaning, such as the count of a specific word in a document, the number of shares of a specific stock held in a portfolio, or the rainfall in a specific hour. It is common to keep a separate list of labels or tags that explain or annotate the meaning of the vector entries. As an example, we might associate the portfolio vector $(100, 50, 20)$ with the list of ticker symbols (AAPL, INTC, AMZN), so we know that assets 1, 2, and 3 are Apple, Intel, and Amazon. In some applications, such as an image, the meaning or ordering of the entries follow known conventions or standards.

1.2 Vector addition

Two vectors *of the same size* can be added together by adding the corresponding elements, to form another vector of the same size, called the *sum* of the vectors. Vector addition is denoted by the symbol +. (Thus the symbol + is overloaded to mean scalar addition when scalars appear on its left- and right-hand sides, and vector addition when vectors appear on its left- and right-hand sides.) For example,

$$\begin{bmatrix} 0 \\ 7 \\ 3 \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 9 \\ 3 \end{bmatrix}.$$

Vector subtraction is similar. As an example,

$$\begin{bmatrix} 1 \\ 9 \end{bmatrix} - \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 8 \end{bmatrix}.$$

The result of vector subtraction is called the *difference* of the two vectors.

Properties. Several properties of vector addition are easily verified. For any vectors a, b, and c of the same size we have the following.

- Vector addition is *commutative*: $a + b = b + a$.
- Vector addition is *associative*: $(a + b) + c = a + (b + c)$. We can therefore write both as $a + b + c$.
- $a + 0 = 0 + a = a$. Adding the zero vector to a vector has no effect. (This is an example where the size of the zero vector follows from the context: It must be the same as the size of a.)
- $a - a = 0$. Subtracting a vector from itself yields the zero vector. (Here too the size of 0 is the size of a.)

To show that these properties hold, we argue using the definition of vector addition and vector equality. As an example, let us show that for any n-vectors a and b, we have $a + b = b + a$. The ith entry of $a + b$ is, by the definition of vector

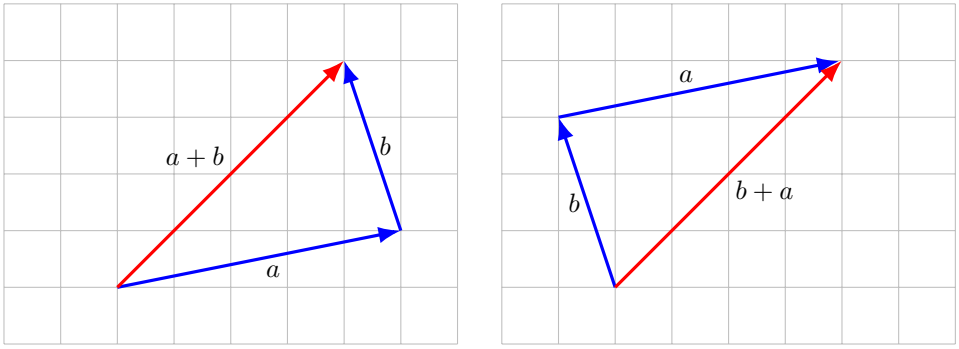

Figure 1.6 *Left.* The lower blue arrow shows the displacement a; the displacement b, shown as the shorter blue arrow, starts from the head of the displacement a and ends at the sum displacement $a + b$, shown as the red arrow. *Right.* The displacement $b + a$.

addition, $a_i + b_i$. The ith entry of $b + a$ is $b_i + a_i$. For any two numbers we have $a_i + b_i = b_i + a_i$, so the ith entries of the vectors $a + b$ and $b + a$ are the same. This is true for all of the entries, so by the definition of vector equality, we have $a + b = b + a$.

Verifying identities like the ones above, and many others we will encounter later, can be tedious. But it is important to understand that the various properties we will list can be derived using elementary arguments like the one above. We recommend that the reader select a few of the properties we will see, and attempt to derive them, just to see that it can be done. (Deriving all of them is overkill.)

Examples.

- *Displacements.* When vectors a and b represent displacements, the sum $a+b$ is the net displacement found by first displacing by a, then displacing by b, as shown in figure 1.6. Note that we arrive at the same vector if we first displace by b and then a. If the vector p represents a position and the vector a represents a displacement, then $p+a$ is the position of the point p, displaced by a, as shown in figure 1.7.

- *Displacements between two points.* If the vectors p and q represent the positions of two points in 2-D or 3-D space, then $p - q$ is the displacement vector from q to p, as illustrated in figure 1.8.

- *Word counts.* If a and b are word count vectors (using the same dictionary) for two documents, the sum $a+b$ is the word count vector of a new document created by combining the original two (in either order). The word count difference vector $a - b$ gives the number of times more each word appears in the first document than the second.

- *Bill of materials.* Suppose $q_1, \ldots, q_N$ are n-vectors that give the quantities of n different resources required to accomplish N tasks. Then the sum n-vector $q_1 + \cdots + q_N$ gives the bill of materials for completing all N tasks.

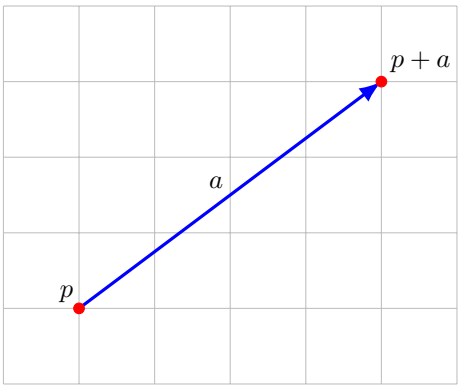

Figure 1.7 The vector $p + a$ is the position of the point represented by p displaced by the displacement represented by a.

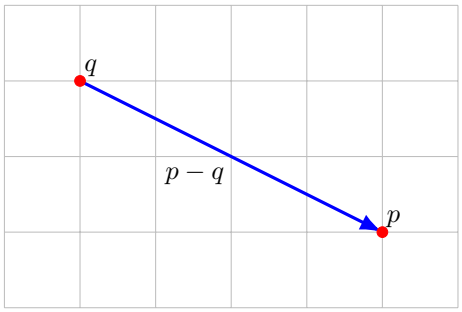

Figure 1.8 The vector $p - q$ represents the displacement from the point represented by q to the point represented by p.

- *Market clearing.* Suppose the n-vector q_i represents the quantities of n goods or resources produced (when positive) or consumed (when negative) by agent i, for $i = 1, \ldots, N$, so $(q_5)_4 = -3.2$ means that agent 5 consumes 3.2 units of resource 4. The sum $s = q_1 + \cdots + q_N$ is the n-vector of total net surplus of the resources (or shortfall, when the entries are negative). When $s = 0$, we have a closed market, which means that the total quantity of each resource produced by the agents balances the total quantity consumed. In other words, the n resources are *exchanged* among the agents. In this case we say that the *market clears* (with the resource vectors $q_1, \ldots, q_N$).

- *Audio addition.* When a and b are vectors representing audio signals over the same period of time, the sum $a + b$ is an audio signal that is perceived as containing both audio signals combined into one. If a represents a recording of a voice, and b a recording of music (of the same length), the audio signal $a+b$ will be perceived as containing both the voice recording and, simultaneously, the music.

- *Feature differences.* If f and g are n-vectors that give n feature values for two items, the difference vector $d = f - g$ gives the difference in feature values for the two objects. For example, $d_7 = 0$ means that the two objects have the same value for feature 7; $d_3 = 1.67$ means that the first object's third feature value exceeds the second object's third feature value by 1.67.

- *Time series.* If a and b represent time series of the same quantity, such as daily profit at two different stores, then $a+b$ represents a time series which is the total daily profit at the two stores. An example (with monthly rainfall) is shown in figure 1.9.

- *Portfolio trading.* Suppose s is an n-vector giving the number of shares of n assets in a portfolio, and b is an n-vector giving the number of shares of the assets that we buy (when b_i is positive) or sell (when b_i is negative). After the asset purchases and sales, our portfolio is given by $s + b$, the sum of the original portfolio vector and the purchase vector b, which is also called the *trade vector* or *trade list*. (The same interpretation works when the portfolio and trade vectors are given in dollar value.)

Addition notation in computer languages. Some computer languages for manipulating vectors define the sum of a vector and a scalar as the vector obtained by adding the scalar to each element of the vector. This is not standard mathematical notation, however, so we will not use it. Even more confusing, in some computer languages the plus symbol is used to denote concatenation of arrays, which means putting one array after another, as in $(1, 2) + (3, 4, 5) = (1, 2, 3, 4, 5)$. While this notation might give a valid expression in some computer languages, it is not standard mathematical notation, and we will not use it in this book. In general, it is very important to distinguish between mathematical notation for vectors (which we use) and the syntax of specific computer languages or software packages for manipulating vectors.

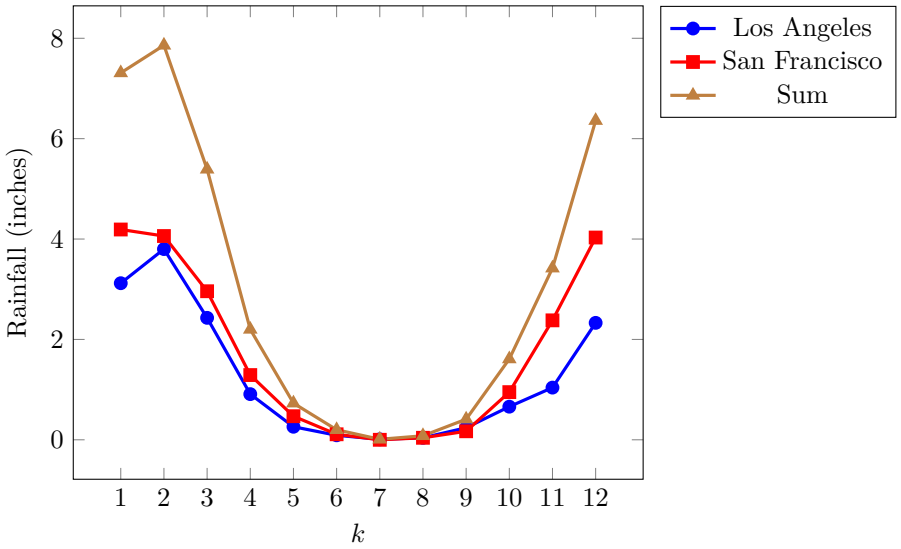

Figure 1.9 Average monthly rainfall in inches measured in downtown Los Angeles and San Francisco International Airport, and their sum. Averages are 30-year averages (1981–2010).

1.3 Scalar-vector multiplication

Another operation is *scalar multiplication* or *scalar-vector multiplication*, in which a vector is multiplied by a scalar (*i.e.*, number), which is done by multiplying every element of the vector by the scalar. Scalar multiplication is denoted by juxtaposition, typically with the scalar on the left, as in

$$(-2)\begin{bmatrix} 1 \\ 9 \\ 6 \end{bmatrix} = \begin{bmatrix} -2 \\ -18 \\ -12 \end{bmatrix}.$$

Scalar-vector multiplication can also be written with the scalar on the right, as in

$$\begin{bmatrix} 1 \\ 9 \\ 6 \end{bmatrix}(1.5) = \begin{bmatrix} 1.5 \\ 13.5 \\ 9 \end{bmatrix}.$$

The meaning is the same: It is the vector obtained by multiplying each element by the scalar. A similar notation is $a/2$, where a is a vector, meaning $(1/2)a$. The scalar-vector product $(-1)a$ is written simply as $-a$. Note that $0\,a = 0$ (where the left-hand zero is the scalar zero, and the right-hand zero is a vector zero of the same size as a).

Properties. By definition, we have $\alpha a = a\alpha$, for any scalar α and any vector a. This is called the *commutative property* of scalar-vector multiplication; it means that scalar-vector multiplication can be written in either order.

Scalar multiplication obeys several other laws that are easy to figure out from the definition. For example, it satisfies the associative property: If a is a vector and β and γ are scalars, we have

$$(\beta\gamma)a = \beta(\gamma a).$$

On the left-hand side we see scalar-scalar multiplication $(\beta\gamma)$ and scalar-vector multiplication; on the right-hand side we see two scalar-vector products. As a consequence, we can write the vector above as $\beta\gamma a$, since it does not matter whether we interpret this as $\beta(\gamma a)$ or $(\beta\gamma)a$.

The associative property holds also when we denote scalar-vector multiplication with the scalar on the right. For example, we have $\beta(\gamma a) = (\beta a)\gamma$, and consequently we can write both as $\beta a\gamma$. As a convention, however, this vector is normally written as $\beta\gamma a$ or as $(\beta\gamma)a$.

If a is a vector and β, γ are scalars, then

$$(\beta + \gamma)a = \beta a + \gamma a.$$

(This is the left-distributive property of scalar-vector multiplication.) Scalar multiplication, like ordinary multiplication, has higher precedence in equations than vector addition, so the right-hand side here, $\beta a+\gamma a$, means $(\beta a)+(\gamma a)$. It is useful to identify the symbols appearing in this formula above. The $+$ symbol on the left is addition of scalars, while the $+$ symbol on the right denotes vector addition. When scalar multiplication is written with the scalar on the right, we have the right-distributive property:

$$a(\beta + \gamma) = a\beta + a\gamma.$$

Scalar-vector multiplication also satisfies another version of the right-distributive property:

$$\beta(a + b) = \beta a + \beta b$$

for any scalar β and any n-vectors a and b. In this equation, both of the $+$ symbols refer to the addition of n-vectors.

Examples.

- *Displacements.* When a vector a represents a displacement, and $\beta > 0$, βa is a displacement in the same direction of a, with its magnitude scaled by β. When $\beta < 0$, βa represents a displacement in the opposite direction of a, with magnitude scaled by $|\beta|$. This is illustrated in figure 1.10.

- *Materials requirements.* Suppose the n-vector q is the bill of materials for producing one unit of some product, *i.e.*, q_i is the amount of raw material required to produce one unit of product. To produce α units of the product will then require raw materials given by αq. (Here we assume that $\alpha \geq 0$.)

- *Audio scaling.* If a is a vector representing an audio signal, the scalar-vector product βa is perceived as the same audio signal, but changed in volume (loudness) by the factor $|\beta|$. For example, when $\beta = 1/2$ (or $\beta = -1/2$), βa is perceived as the same audio signal, but quieter.

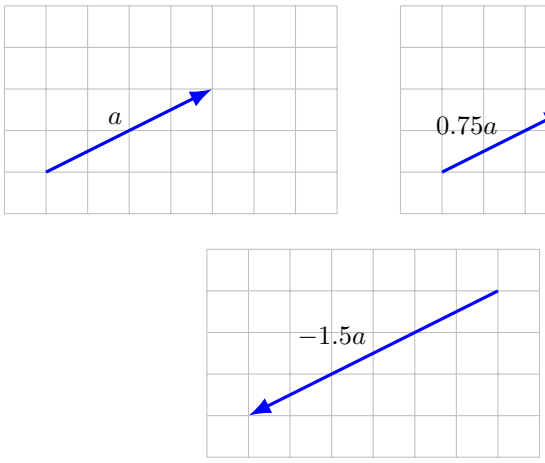

Figure 1.10 The vector $0.75a$ represents the displacement in the direction of the displacement a, with magnitude scaled by 0.75; $(-1.5)a$ represents the displacement in the opposite direction, with magnitude scaled by 1.5.

Linear combinations. If $a_1, \ldots, a_m$ are n-vectors, and $\beta_1, \ldots, \beta_m$ are scalars, the n-vector

$$\beta_1 a_1 + \cdots + \beta_m a_m$$

is called a *linear combination* of the vectors $a_1, \ldots, a_n$. The scalars $\beta_1, \ldots, \beta_m$ are called the *coefficients* of the linear combination.

Linear combination of unit vectors. We can write any n-vector b as a linear combination of the standard unit vectors, as

$$b = b_1 e_1 + \cdots + b_n e_n. \tag{1.1}$$

In this equation b_i is the ith entry of b (*i.e.*, a scalar), and e_i is the ith unit vector. In the linear combination of $e_1, \ldots, e_n$ given in (1.1), the coefficients are the entries of the vector b. A specific example is

$$\begin{bmatrix} -1 \\ 3 \\ 5 \end{bmatrix} = (-1) \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + 3 \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + 5 \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}.$$

Special linear combinations. Some linear combinations of the vectors $a_1, \ldots, a_m$ have special names. For example, the linear combination with $\beta_1 = \cdots = \beta_m = 1$, given by $a_1 + \cdots + a_m$, is the *sum* of the vectors, and the linear combination with $\beta_1 = \cdots = \beta_m = 1/m$, given by $(1/m)(a_1 + \cdots + a_m)$, is the *average* of the vectors. When the coefficients sum to one, *i.e.*, $\beta_1 + \cdots + \beta_m = 1$, the linear combination is called an *affine combination*. When the coefficients in an affine combination are nonnegative, it is called a *convex combination*, a *mixture*, or a *weighted average*. The coefficients in an affine or convex combination are sometimes given as percentages, which add up to 100%.

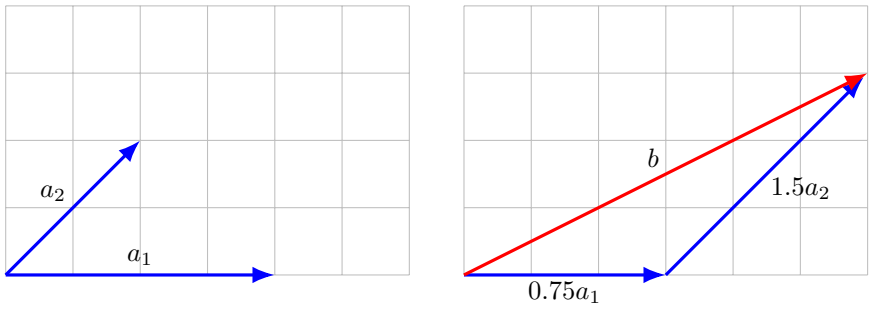

Figure 1.11 *Left.* Two 2-vectors a_1 and a_2. *Right.* The linear combination $b = 0.75a_1 + 1.5a_2$.

Examples.

- *Displacements.* When the vectors represent displacements, a linear combination is the sum of the scaled displacements. This is illustrated in figure 1.11.

- *Audio mixing.* When $a_1, \ldots, a_m$ are vectors representing audio signals (over the same period of time, for example, simultaneously recorded), they are called *tracks*. The linear combination $\beta_1 a_1 + \cdots + \beta_m a_m$ is perceived as a mixture (also called a *mix*) of the audio tracks, with relative loudness given by $|\beta_1|, \ldots, |\beta_m|$. A producer in a studio, or a sound engineer at a live show, chooses values of $\beta_1, \ldots, \beta_m$ to give a good balance between the different instruments, vocals, and drums.

- *Cash flow replication.* Suppose that $c_1, \ldots, c_m$ are vectors that represent cash flows, such as particular types of loans or investments. The linear combination $f = \beta_1 c_1 + \cdots + \beta_m c_m$ represents another cash flow. We say that the cash flow f has been *replicated* by the (linear combination of the) original cash flows $c_1, \ldots, c_m$. As an example, $c_1 = (1, -1.1, 0)$ represents a \$1 loan from period 1 to period 2 with 10% interest, and $c_2 = (0, 1, -1.1)$ represents a \$1 loan from period 2 to period 3 with 10% interest. The linear combination

  $$d = c_1 + 1.1c_2 = (1, 0, -1.21)$$

  represents a two period loan of \$1 in period 1, with compounded 10% interest. Here we have replicated a two period loan from two one period loans.

- *Line and segment.* When a and b are different n-vectors, the affine combination $c = (1 - \theta)a + \theta b$, where θ is a scalar, describes a point on the *line* passing through a and b. When $0 \leq \theta \leq 1$, c is a convex combination of a and b, and is said to lie on the *segment* between a and b. For $n = 2$ and $n = 3$, with the vectors representing coordinates of 2-D or 3-D points, this agrees with the usual geometric notion of line and segment. But we can also talk about the line passing through two vectors of dimension 100. This is illustrated in figure 1.12.

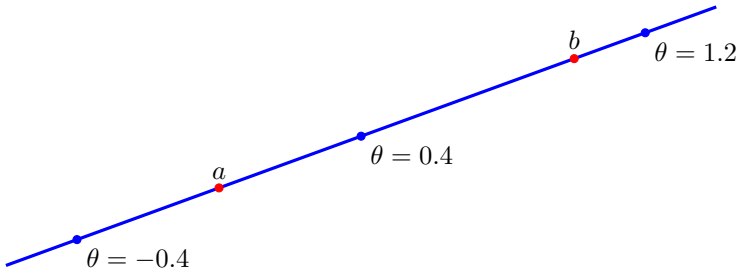

Figure 1.12 The affine combination $(1-\theta)a+\theta b$ for different values of θ. These points are on the line passing through a and b; for θ between 0 and 1, the points are on the line segment between a and b.

1.4 Inner product

The (standard) *inner product* (also called *dot product*) of two n-vectors is defined as the scalar

$$a^T b = a_1b_1 + a_2b_2 + \cdots + a_nb_n,$$

the sum of the products of corresponding entries. (The origin of the superscript 'T' in the inner product notation a^Tb will be explained in chapter 6.) Some other notations for the inner product (that we will not use in this book) are $\langle a, b\rangle$, $\langle a|b\rangle$, (a, b), and $a \cdot b$. (In the notation used in this book, (a, b) denotes a stacked vector of length $2n$.) As you might guess, there is also a vector *outer product*, which we will encounter later, in §10.1. As a specific example of the inner product, we have

$$\begin{bmatrix} -1 \\ 2 \\ 2 \end{bmatrix}^T \begin{bmatrix} 1 \\ 0 \\ -3 \end{bmatrix} = (-1)(1) + (2)(0) + (2)(-3) = -7.$$

When $n = 1$, the inner product reduces to the usual product of two numbers.

Properties. The inner product satisfies some simple properties that are easily verified from the definition. If a, b, and c are vectors of the same size, and γ is a scalar, we have the following.

- *Commutativity.* $a^Tb = b^Ta$. The order of the two vector arguments in the inner product does not matter.
- *Associativity with scalar multiplication.* $(\gamma a)^Tb = \gamma(a^Tb)$, so we can write both as γa^Tb.
- *Distributivity with vector addition.* $(a+b)^Tc = a^Tc+b^Tc$. The inner product can be distributed across vector addition.

These can be combined to obtain other identities, such as $a^T(\gamma b) = \gamma(a^Tb)$, or $a^T(b+\gamma c) = a^Tb+\gamma a^Tc$. As another useful example, we have, for any vectors a, b, c, d of the same size,

$$(a+b)^T(c+d) = a^Tc + a^Td + b^Tc + b^Td.$$

This formula expresses an inner product on the left-hand side as a sum of four inner products on the right-hand side, and is analogous to expanding a product of sums in algebra. Note that on the left-hand side, the two addition symbols refer to vector addition, whereas on the right-hand side, the three addition symbols refer to scalar (number) addition.

General examples.

- *Unit vector.* $e_i^T a = a_i$. The inner product of a vector with the ith standard unit vector gives (or 'picks out') the ith element a.
- *Sum.* $\mathbf{1}^T a = a_1 + \cdots + a_n$. The inner product of a vector with the vector of ones gives the sum of the elements of the vector.
- *Average.* $(\mathbf{1}/n)^T a = (a_1 + \cdots + a_n)/n$. The inner product of an n-vector with the vector $\mathbf{1}/n$ gives the average or mean of the elements of the vector. The average of the entries of a vector is denoted by $\mathbf{avg}(x)$. The Greek letter μ is a traditional symbol used to denote the average or mean.
- *Sum of squares.* $a^T a = a_1^2 + \cdots + a_n^2$. The inner product of a vector with itself gives the sum of the squares of the elements of the vector.
- *Selective sum.* Let b be a vector all of whose entries are either 0 or 1. Then $b^T a$ is the sum of the elements in a for which $b_i = 1$.

Block vectors. If the vectors a and b are block vectors, and the corresponding blocks have the same sizes (in which case we say they *conform*), then we have

$$a^T b = \begin{bmatrix} a_1 \\ \vdots \\ a_k \end{bmatrix}^T \begin{bmatrix} b_1 \\ \vdots \\ b_k \end{bmatrix} = a_1^T b_1 + \cdots + a_k^T b_k.$$

The inner product of block vectors is the sum of the inner products of the blocks.

Applications. The inner product is useful in many applications, a few of which we list here.

- *Co-occurrence.* If a and b are n-vectors that describe occurrence, *i.e.*, each of their elements is either 0 or 1, then $a^T b$ gives the total number of indices for which a_i and b_i are both one, that is, the total number of co-occurrences. If we interpret the vectors a and b as describing subsets of n objects, then $a^T b$ gives the number of objects in the intersection of the two subsets. This is illustrated in figure 1.13, for two subsets A and B of 7 objects, labeled $1, \ldots, 7$, with corresponding occurrence vectors

  $$a = (0, 1, 1, 1, 1, 1, 1), \qquad b = (1, 0, 1, 0, 1, 0, 0).$$

  Here we have $a^T b = 2$, which is the number of objects in both A and B (*i.e.*, objects 3 and 5).

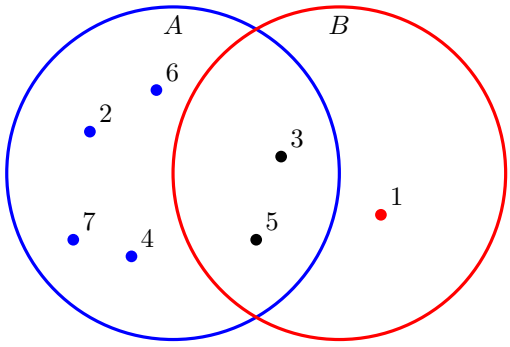

Figure 1.13 Two sets A and B, containing seven objects.

- *Weights, features, and score.* When the vector f represents a set of features of an object, and w is a vector of the same size (often called a *weight vector*), the inner product $w^T f$ is the sum of the feature values, scaled (or weighted) by the weights, and is sometimes called a *score*. For example, if the features are associated with a loan applicant (*e.g.*, age, income, ...), we might interpret $s = w^T f$ as a credit score. In this example we can interpret w_i as the weight given to feature i in forming the score.

- *Price-quantity.* If p represents a vector of prices of n goods, and q is a vector of quantities of the n goods (say, the bill of materials for a product), then their inner product $p^T q$ is the total cost of the goods given by the vector q.

- *Speed-time.* A vehicle travels over n segments with constant speed in each segment. Suppose the n-vector s gives the speed in the segments, and the n-vector t gives the times taken to traverse the segments. Then $s^T t$ is the total distance traveled.

- *Probability and expected values.* Suppose the n-vector p has nonnegative entries that sum to one, so it describes a set of proportions among n items, or a set of probabilities of n outcomes, one of which must occur. Suppose f is another n-vector, where we interpret f_i as the value of some quantity if outcome i occurs. Then $f^T p$ gives the expected value or mean of the quantity, under the probabilities (or fractions) given by p.

- *Polynomial evaluation.* Suppose the n-vector c represents the coefficients of a polynomial p of degree $n-1$ or less:

$$p(x) = c_1 + c_2 x + \cdots + c_{n-1} x^{n-2} + c_n x^{n-1}.$$

  Let t be a number, and let $z = (1, t, t^2, \ldots, t^{n-1})$ be the n-vector of powers of t. Then $c^T z = p(t)$, the value of the polynomial p at the point t. So the inner product of a polynomial coefficient vector and vector of powers of a number evaluates the polynomial at the number.

- *Discounted total.* Let c be an n-vector representing a cash flow, with c_i the cash received (when $c_i > 0$) in period i. Let d be the n-vector defined as

$$d = (1, 1/(1+r), \ldots, 1/(1+r)^{n-1}),$$

  where $r \geq 0$ is an interest rate. Then

$$d^T c = c_1 + c_2/(1+r) + \cdots + c_n/(1+r)^{n-1}$$

  is the discounted total of the cash flow, *i.e.*, its *net present value* (NPV), with interest rate r.

- *Portfolio value.* Suppose s is an n-vector representing the holdings in shares of a portfolio of n different assets, with negative values meaning short positions. If p is an n-vector giving the prices of the assets, then $p^T s$ is the total (or net) value of the portfolio.

- *Portfolio return.* Suppose r is the vector of (fractional) returns of n assets over some time period, *i.e.*, the asset relative price changes

$$r_i = \frac{p_i^{\text{final}} - p_i^{\text{initial}}}{p_i^{\text{initial}}}, \quad i = 1, \ldots, n,$$

  where p_i^{initial} and p_i^{final} are the (positive) prices of asset i at the beginning and end of the investment period. If h is an n-vector giving our portfolio, with h_i denoting the dollar value of asset i held, then the inner product $r^T h$ is the total return of the portfolio, in dollars, over the period. If w represents the fractional (dollar) holdings of our portfolio, then $r^T w$ gives the total return of the portfolio. For example, if $r^T w = 0.09$, then our portfolio return is 9%. If we had invested \$10000 initially, we would have earned \$900.

- *Document sentiment analysis.* Suppose the n-vector x represents the histogram of word occurrences in a document, from a dictionary of n words. Each word in the dictionary is assigned to one of three sentiment categories: *Positive*, *Negative*, and *Neutral*. The list of positive words might include 'nice' and 'superb'; the list of negative words might include 'bad' and 'terrible'. Neutral words are those that are neither positive nor negative. We encode the word categories as an n-vector w, with $w_i = 1$ if word i is positive, with $w_i = -1$ if word i is negative, and $w_i = 0$ if word i is neutral. The number $w^T x$ gives a (crude) measure of the sentiment of the document.

1.5 Complexity of vector computations

Computer representation of numbers and vectors. Real numbers are stored in computers using *floating point format*, which represents a real number using a block of 64 *bits* (0s and 1s), or 8 *bytes* (groups of 8 bits). Each of the 2^{64} possible sequences of bits corresponds to a specific real number. The floating point numbers

span a very wide range of values, and are very closely spaced, so numbers that arise in applications can be approximated as floating point numbers to an accuracy of around 10 digits, which is good enough for almost all practical applications. Integers are stored in a more compact format, and are represented exactly.

Vectors are stored as arrays of floating point numbers (or integers, when the entries are all integers). Storing an n-vector requires $8n$ bytes to store. Current memory and storage devices, with capacities measured in many gigabytes (10^9 bytes), can easily store vectors with dimensions in the millions or billions. Sparse vectors are stored in a more efficient way that keeps track of indices and values of the nonzero entries.

Floating point operations. When computers carry out addition, subtraction, multiplication, division, or other arithmetic operations on numbers represented in floating point format, the result is rounded to the nearest floating point number. These operations are called *floating point operations*. The very small error in the computed result is called (floating point) *round-off error*. In most applications, these very small errors have no practical effect. Floating point round-off errors, and methods to mitigate their effect, are studied in a field called *numerical analysis*. In this book we will not consider floating point round-off error, but you should be aware that it exists. For example, when a computer evaluates the left-hand and right-hand sides of a mathematical identity, you should not be surprised if the two numbers are not equal. They should, however, be very close.

Flop counts and complexity. So far we have seen only a few vector operations, like scalar multiplication, vector addition, and the inner product. How quickly these operations can be carried out by a computer depends very much on the computer hardware and software, and the size of the vector.

A very rough estimate of the time required to carry out some computation, such as an inner product, can be found by counting the total number of floating point operations, or FLOPs. This term is in such common use that the acronym is now written in lower case letters, as flops, and the speed with which a computer can carry out flops is expressed in Gflop/s (gigaflops per second, *i.e.*, billions of flops per second). Typical current values are in the range of 1–10 Gflop/s, but this can vary by several orders of magnitude. The actual time it takes a computer to carry out some computation depends on many other factors beyond the total number of flops required, so time estimates based on counting flops are very crude, and are not meant to be more accurate than a factor of ten or so. For this reason, gross approximations (such as ignoring a factor of 2) can be used when counting the flops required in a computation.

The *complexity* of an operation is the number of flops required to carry it out, as a function of the size or sizes of the input to the operation. Usually the complexity is highly simplified, dropping terms that are small or negligible (compared to other terms) when the sizes of the inputs are large. In theoretical computer science, the term 'complexity' is used in a different way, to mean the number of flops of the best method to carry out the computation, *i.e.*, the one that requires the fewest flops. In this book, we use the term complexity to mean the number of flops required by a specific method.

Complexity of vector operations. Scalar-vector multiplication ax, where x is an n-vector, requires n multiplications, *i.e.*, ax_i for $i = 1, \ldots, n$. Vector addition $x+y$ of two n-vectors takes n additions, *i.e.*, $x_i + y_i$ for $i = 1, \ldots, n$. Computing the inner product $x^T y = x_1 y_1 + \cdots + x_n y_n$ of two n-vectors takes $2n-1$ flops, n scalar multiplications and $n-1$ scalar additions. So scalar multiplication, vector addition, and the inner product of n-vectors require n, n, and $2n-1$ flops, respectively. We only need an estimate, so we simplify the last to $2n$ flops, and say that the *complexity* of scalar multiplication, vector addition, and the inner product of n-vectors is n, n, and $2n$ flops, respectively. We can guess that a 1 Gflop/s computer can compute the inner product of two vectors of size one million in around one thousandth of a second, but we should not be surprised if the actual time differs by a factor of 10 from this value.

The *order* of the computation is obtained by ignoring any constant that multiplies a power of the dimension. So we say that the three vector operations scalar multiplication, vector addition, and inner product have order n. Ignoring the factor of 2 dropped in the actual complexity of the inner product is reasonable, since we do not expect flop counts to predict the running time with an accuracy better than a factor of 2. The order is useful in understanding how the time to execute the computation will scale when the size of the operands changes. An order n computation should take around 10 times longer to carry out its computation on an input that is 10 times bigger.

Complexity of sparse vector operations. If x is sparse, then computing ax requires $\mathbf{nnz}(x)$ flops. If x and y are sparse, computing $x+y$ requires no more than $\min\{\mathbf{nnz}(x), \mathbf{nnz}(y)\}$ flops (since no arithmetic operations are required to compute $(x+y)_i$ when either x_i or y_i is zero). If the sparsity patterns of x and y do not overlap (intersect), then zero flops are needed to compute $x+y$. The inner product calculation is similar: computing $x^T y$ requires no more than $2\min\{\mathbf{nnz}(x), \mathbf{nnz}(y)\}$ flops. When the sparsity patterns of x and y do not overlap, computing $x^T y$ requires zero flops, since $x^T y = 0$ in this case.

Exercises

1.1 *Vector equations.* Determine whether each of the equations below is true, false, or contains bad notation (and therefore does not make sense).

(a) $\begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} = (1, 2, 1)$.

(b) $\begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 1, & 2, & 1 \end{bmatrix}$.

(c) $(1, (2, 1)) = ((1, 2), 1)$.

1.2 *Vector notation.* Which of the following expressions uses correct notation? When the expression does make sense, give its length. In the following, a and b are 10-vectors, and c is a 20-vector.

(a) $a + b - c_{3:12}$.

(b) $(a, b, c_{3:13})$.

(c) $2a + c$.

(d) $(a, 1) + (c_1, b)$.

(e) $((a, b), a)$.

(f) $\begin{bmatrix} a & b \end{bmatrix} + 4c$.

(g) $\begin{bmatrix} a \\ b \end{bmatrix} + 4c$.

1.3 *Overloading.* Which of the following expressions uses correct notation? If the notation is correct, is it also unambiguous? Assume that a is a 10-vector and b is a 20-vector.

(a) $b = (0, a)$.

(b) $a = (0, b)$.

(c) $b = (0, a, 0)$.

(d) $a = 0 = b$.

1.4 *Periodic energy usage.* The 168-vector w gives the hourly electricity consumption of a manufacturing plant, starting on Sunday midnight to 1AM, over one week, in MWh (megawatt-hours). The consumption pattern is the same each day, *i.e.*, it is 24-periodic, which means that $w_{t+24} = w_t$ for $t = 1, \ldots, 144$. Let d be the 24-vector that gives the energy consumption over one day, starting at midnight.

(a) Use vector notation to express w in terms of d.

(b) Use vector notation to express d in terms of w.

1.5 *Interpreting sparsity.* Suppose the n-vector x is sparse, *i.e.*, has only a few nonzero entries. Give a short sentence or two explaining what this means in each of the following contexts.

(a) x represents the daily cash flow of some business over n days.

(b) x represents the annual dollar value purchases by a customer of n products or services.

(c) x represents a portfolio, say, the dollar value holdings of n stocks.

(d) x represents a bill of materials for a project, *i.e.*, the amounts of n materials needed.

(e) x represents a monochrome image, *i.e.*, the brightness values of n pixels.

(f) x is the daily rainfall in a location over one year.

1.6 *Vector of differences.* Suppose x is an n-vector. The associated vector of differences is the $(n-1)$-vector d given by $d = (x_2 - x_1, x_3 - x_2, \ldots, x_n - x_{n-1})$. Express d in terms of x using vector operations (*e.g.*, slicing notation, sum, difference, linear combinations, inner product). The difference vector has a simple interpretation when x represents a time series. For example, if x gives the daily value of some quantity, d gives the day-to-day changes in the quantity.

1.7 *Transforming between two encodings for Boolean vectors.* A Boolean n-vector is one for which all entries are either 0 or 1. Such vectors are used to encode whether each of n conditions holds, with $a_i = 1$ meaning that condition i holds. Another common encoding of the same information uses the two values -1 and $+1$ for the entries. For example the Boolean vector $(0, 1, 1, 0)$ would be written using this alternative encoding as $(-1, +1, +1, -1)$. Suppose that x is a Boolean vector with entries that are 0 or 1, and y is a vector encoding the same information using the values -1 and $+1$. Express y in terms of x using vector notation. Also, express x in terms of y using vector notation.

1.8 *Profit and sales vectors.* A company sells n different products or items. The n-vector p gives the profit, in dollars per unit, for each of the n items. (The entries of p are typically positive, but a few items might have negative entries. These items are called *loss leaders*, and are used to increase customer engagement in the hope that the customer will make other, profitable purchases.) The n-vector s gives the total sales of each of the items, over some period (such as a month), *i.e.*, s_i is the total number of units of item i sold. (These are also typically nonnegative, but negative entries can be used to reflect items that were purchased in a previous time period and returned in this one.) Express the total profit in terms of p and s using vector notation.

1.9 *Symptoms vector.* A 20-vector s records whether each of 20 different symptoms is present in a medical patient, with $s_i = 1$ meaning the patient has the symptom and $s_i = 0$ meaning she does not. Express the following using vector notation.

(a) The total number of symptoms the patient has.

(b) The patient exhibits five out of the first ten symptoms.

1.10 *Total score from course record.* The record for each student in a class is given as a 10-vector r, where $r_1, \ldots, r_8$ are the grades for the 8 homework assignments, each on a 0–10 scale, r_9 is the midterm exam grade on a 0–120 scale, and r_{10} is final exam score on a 0–160 scale. The student's total course score s, on a 0–100 scale, is based 25% on the homework, 35% on the midterm exam, and 40% on the final exam. Express s in the form $s = w^T r$. (That is, determine the 10-vector w.) You can give the coefficients of w to 4 digits after the decimal point.

1.11 *Word count and word count histogram vectors.* Suppose the n-vector w is the word count vector associated with a document and a dictionary of n words. For simplicity we will assume that all words in the document appear in the dictionary.

(a) What is $\mathbf{1}^T w$?

(b) What does $w_{282} = 0$ mean?

(c) Let h be the n-vector that gives the histogram of the word counts, *i.e.*, h_i is the fraction of the words in the document that are word i. Use vector notation to express h in terms of w. (You can assume that the document contains at least one word.)

1.12 *Total cash value.* An international company holds cash in five currencies: USD (US dollar), RMB (Chinese yuan), EUR (euro), GBP (British pound), and JPY (Japanese yen), in amounts given by the 5-vector c. For example, c_2 gives the number of RMB held. Negative entries in c represent liabilities or amounts owed. Express the total (net) value of the cash in USD, using vector notation. Be sure to give the size and define the entries of any vectors that you introduce in your solution. Your solution can refer to currency exchange rates.

1.13 *Average age in a population.* Suppose the 100-vector x represents the distribution of ages in some population of people, with x_i being the number of $i-1$ year olds, for $i = 1, \ldots, 100$. (You can assume that $x \neq 0$, and that there is no one in the population over age 99.) Find expressions, using vector notation, for the following quantities.

(a) The total number of people in the population.

(b) The total number of people in the population age 65 and over.

(c) The average age of the population. (You can use ordinary division of numbers in your expression.)

1.14 *Industry or sector exposure.* Consider a set of n assets or stocks that we invest in. Let f be an n-vector that encodes whether each asset is in some specific industry or sector, *e.g.*, pharmaceuticals or consumer electronics. Specifically, we take $f_i = 1$ if asset i is in the sector, and $f_i = 0$ if it is not. Let the n-vector h denote a portfolio, with h_i the dollar value held in asset i (with negative meaning a short position). The inner product $f^T h$ is called the (dollar value) *exposure* of our portfolio to the sector. It gives the net dollar value of the portfolio that is invested in assets from the sector. A portfolio h is called *neutral* (to a sector or industry) if $f^T h = 0$.

A portfolio h is called *long only* if each entry is nonnegative, *i.e.*, $h_i \geq 0$ for each i. This means the portfolio does not include any short positions.

What does it mean if a long-only portfolio is neutral to a sector, say, pharmaceuticals? Your answer should be in simple English, but you should back up your conclusion with an argument.

1.15 *Cheapest supplier.* You must buy n raw materials in quantities given by the n-vector q, where q_i is the amount of raw material i that you must buy. A set of K potential suppliers offer the raw materials at prices given by the n-vectors $p_1, \ldots, p_K$. (Note that p_k is an n-vector; $(p_k)_i$ is the price that supplier k charges per unit of raw material i.) We will assume that all quantities and prices are positive.

If you must choose just one supplier, how would you do it? Your answer should use vector notation.

A (highly paid) consultant tells you that you might do better (*i.e.*, get a better total cost) by splitting your order into two, by choosing two suppliers and ordering $(1/2)q$ (*i.e.*, half the quantities) from each of the two. He argues that having a diversity of suppliers is better. Is he right? If so, explain how to find the two suppliers you would use to fill half the order.

1.16 *Inner product of nonnegative vectors.* A vector is called *nonnegative* if all its entries are nonnegative.

(a) Explain why the inner product of two nonnegative vectors is nonnegative.

(b) Suppose the inner product of two nonnegative vectors is zero. What can you say about them? Your answer should be in terms of their respective sparsity patterns, *i.e.*, which entries are zero and nonzero.

1.17 *Linear combinations of cash flows.* We consider cash flow vectors over T time periods, with a positive entry meaning a payment received, and negative meaning a payment made. A (unit) *single period loan*, at time period t, is the T-vector l_t that corresponds to a payment received of \$1 in period t and a payment made of $\$(1 + r)$ in period $t + 1$, with all other payments zero. Here $r > 0$ is the interest rate (over one period).

Let c be a \$1 $T - 1$ period loan, starting at period 1. This means that \$1 is received in period 1, $\$(1 + r)^{T-1}$ is paid in period T, and all other payments (*i.e.*, $c_2, \ldots, c_{T-1}$) are zero. Express c as a linear combination of single period loans.

1.18 *Linear combinations of linear combinations.* Suppose that each of the vectors $b_1, \ldots, b_k$ is a linear combination of the vectors $a_1, \ldots, a_m$, and c is a linear combination of $b_1, \ldots, b_k$. Then c is a linear combination of $a_1, \ldots, a_m$. Show this for the case with $m = k = 2$. (Showing it in general is not much more difficult, but the notation gets more complicated.)

1.19 *Auto-regressive model.* Suppose that $z_1, z_2, \ldots$ is a time series, with the number z_t giving the value in period or time t. For example z_t could be the gross sales at a particular store on day t. An *auto-regressive* (AR) model is used to predict z_{t+1} from the previous M values, $z_t, z_{t-1}, \ldots, z_{t-M+1}$:

$$\hat{z}_{t+1} = (z_t, z_{t-1}, \ldots, z_{t-M+1})^T \beta, \quad t = M, M+1, \ldots.$$

Here $\hat{z}_{t+1}$ denotes the AR model's prediction of z_{t+1}, M is the memory length of the AR model, and the M-vector β is the AR model coefficient vector. For this problem we will assume that the time period is daily, and $M = 10$. Thus, the AR model predicts tomorrow's value, given the values over the last 10 days.

For each of the following cases, give a short interpretation or description of the AR model in English, without referring to mathematical concepts like vectors, inner product, and so on. You can use words like 'yesterday' or 'today'.

(a) $\beta \approx e_1$.

(b) $\beta \approx 2e_1 - e_2$.

(c) $\beta \approx e_6$.

(d) $\beta \approx 0.5e_1 + 0.5e_2$.

1.20 How many bytes does it take to store 100 vectors of length 10^5? How many flops does it take to form a linear combination of them (with 100 nonzero coefficients)? About how long would this take on a computer capable of carrying out 1 Gflop/s?

Chapter 2

Linear functions

In this chapter we introduce linear and affine functions, and describe some common settings where they arise, including regression models.

2.1 Linear functions

Function notation. The notation $f : \mathbf{R}^n \to \mathbf{R}$ means that f is a *function* that maps real n-vectors to real numbers, *i.e.*, it is a scalar-valued function of n-vectors. If x is an n-vector, then $f(x)$, which is a scalar, denotes the *value* of the function f at x. (In the notation $f(x)$, x is referred to as the *argument* of the function.) We can also interpret f as a function of n scalar arguments, the entries of the vector argument, in which case we write $f(x)$ as

$$f(x) = f(x_1, x_2, \ldots, x_n).$$

Here we refer to $x_1, \ldots, x_n$ as the arguments of f. We sometimes say that f is real-valued, or scalar-valued, to emphasize that $f(x)$ is a real number or scalar.

To describe a function $f : \mathbf{R}^n \to \mathbf{R}$, we have to specify what its value is for any possible argument $x \in \mathbf{R}^n$. For example, we can define a function $f : \mathbf{R}^4 \to \mathbf{R}$ by

$$f(x) = x_1 + x_2 - x_4^2$$

for any 4-vector x. In words, we might describe f as the sum of the first two elements of its argument, minus the square of the last entry of the argument. (This particular function does not depend on the third element of its argument.)

Sometimes we introduce a function without formally assigning a symbol for it, by directly giving a formula for its value in terms of its arguments, or describing how to find its value from its arguments. An example is the *sum function*, whose value is $x_1 + \cdots + x_n$. We can give a name to the value of the function, as in $y = x_1 + \cdots + x_n$, and say that y is a function of x, in this case, the sum of its entries.

Many functions are not given by formulas or equations. As an example, suppose $f : \mathbf{R}^3 \to \mathbf{R}$ is the function that gives the lift (vertical upward force) on a particular

airplane, as a function of the 3-vector x, where x_1 is the angle of attack of the airplane (*i.e.*, the angle between the airplane body and its direction of motion), x_2 is its air speed, and x_3 is the air density.

The inner product function. Suppose a is an n-vector. We can define a scalar-valued function f of n-vectors, given by

$$f(x) = a^T x = a_1x_1 + a_2x_2 + \cdots + a_nx_n \tag{2.1}$$

for any n-vector x. This function gives the inner product of its n-vector argument x with some (fixed) n-vector a. We can also think of f as forming a weighted sum of the elements of x; the elements of a give the weights used in forming the weighted sum.

Superposition and linearity. The inner product function f defined in (2.1) satisfies the property

$$\begin{aligned} f(\alpha x + \beta y) &= a^T(\alpha x + \beta y) \\ &= a^T(\alpha x) + a^T(\beta y) \\ &= \alpha(a^T x) + \beta(a^T y) \\ &= \alpha f(x) + \beta f(y) \end{aligned}$$

for all n-vectors x, y, and all scalars α, β. This property is called *superposition*. A function that satisfies the superposition property is called *linear*. We have just shown that the inner product with a fixed vector is a linear function.

The superposition equality

$$f(\alpha x + \beta y) = \alpha f(x) + \beta f(y) \tag{2.2}$$

looks deceptively simple; it is easy to read it as just a re-arrangement of the parentheses and the order of a few terms. But in fact it says a lot. On the left-hand side, the term $\alpha x + \beta y$ involves *scalar-vector* multiplication and *vector addition*. On the right-hand side, $\alpha f(x) + \beta f(y)$ involves ordinary *scalar multiplication* and *scalar addition*.

If a function f is linear, superposition extends to linear combinations of any number of vectors, and not just linear combinations of two vectors: We have

$$f(\alpha_1x_1 + \cdots + \alpha_kx_k) = \alpha_1f(x_1) + \cdots + \alpha_kf(x_k),$$

for any n vectors $x_1, \ldots, x_k$, and any scalars $\alpha_1, \ldots, \alpha_k$. (This more general k-term form of superposition reduces to the two-term form given above when $k = 2$.) To see this, we note that

$$\begin{aligned} f(\alpha_1x_1 + \cdots + \alpha_kx_k) &= \alpha_1f(x_1) + f(\alpha_2x_2 + \cdots + \alpha_kx_k) \\ &= \alpha_1f(x_1) + \alpha_2f(x_2) + f(\alpha_3x_3 + \cdots + \alpha_kx_k) \\ &\vdots \\ &= \alpha_1f(x_1) + \cdots + \alpha_kf(x_k). \end{aligned}$$

In the first line here, we apply (two-term) superposition to the argument

$$\alpha_1 x_1 + (1)(\alpha_2 x_2 + \cdots + \alpha_k x_k),$$

and in the other lines we apply this recursively.

The superposition equality (2.2) is sometimes broken down into two properties, one involving the scalar-vector product and one involving vector addition in the argument. A function $f : \mathbf{R}^n \to \mathbf{R}$ is linear if it satisfies the following two properties.

- *Homogeneity.* For any n-vector x and any scalar α, $f(\alpha x) = \alpha f(x)$.
- *Additivity.* For any n-vectors x and y, $f(x + y) = f(x) + f(y)$.

Homogeneity states that scaling the (vector) argument is the same as scaling the function value; additivity says that adding (vector) arguments is the same as adding the function values.

Inner product representation of a linear function. We saw above that a function defined as the inner product of its argument with some fixed vector is linear. The converse is also true: If a function is linear, then it can be expressed as the inner product of its argument with some fixed vector.

Suppose f is a scalar-valued function of n-vectors, and is linear, *i.e.*, (2.2) holds for all n-vectors x, y, and all scalars α, β. Then there is an n-vector a such that $f(x) = a^T x$ for all x. We call $a^T x$ the *inner product representation* of f.

To see this, we use the identity (1.1) to express an arbitrary n-vector x as $x = x_1 e_1 + \cdots + x_n e_n$. If f is linear, then by multi-term superposition we have

$$\begin{aligned} f(x) &= f(x_1 e_1 + \cdots + x_n e_n) \\ &= x_1 f(e_1) + \cdots + x_n f(e_n) \\ &= a^T x, \end{aligned}$$

with $a = (f(e_1), f(e_2), \ldots, f(e_n))$. The formula just derived,

$$f(x) = x_1 f(e_1) + x_2 f(e_2) + \cdots + x_n f(e_n) \tag{2.3}$$

which holds for any linear scalar-valued function f, has several interesting implications. Suppose, for example, that the linear function f is given as a subroutine (or a physical system) that computes (or results in the output) $f(x)$ when we give the argument (or input) x. Once we have found $f(e_1)$, ..., $f(e_n)$, by n calls to the subroutine (or n experiments), we can predict (or simulate) what $f(x)$ will be, for *any* vector x, using the formula (2.3).

The representation of a linear function f as $f(x) = a^T x$ is *unique*, which means that there is only one vector a for which $f(x) = a^T x$ holds for all x. To see this, suppose that we have $f(x) = a^T x$ for all x, and also $f(x) = b^T x$ for all x. Taking $x = e_i$, we have $f(e_i) = a^T e_i = a_i$, using the formula $f(x) = a^T x$. Using the formula $f(x) = b^T x$, we have $f(e_i) = b^T e_i = b_i$. These two numbers must be the same, so we have $a_i = b_i$. Repeating this argument for $i = 1, \ldots, n$, we conclude that the corresponding elements in a and b are the same, so $a = b$.

Examples.

- *Average.* The *mean* or *average* value of an n-vector is defined as

$$f(x) = (x_1 + x_2 + \cdots + x_n)/n,$$

  and is denoted $\mathbf{avg}(x)$ (and sometimes $\overline{x}$). The average of a vector is a linear function. It can be expressed as $\mathbf{avg}(x) = a^T x$ with

$$a = (1/n, \ldots, 1/n) = \mathbf{1}/n.$$

- *Maximum.* The maximum element of an n-vector x, $f(x) = \max\{x_1, \ldots, x_n\}$, is not a linear function (except when $n = 1$). We can show this by a counterexample for $n = 2$. Take $x = (1, -1)$, $y = (-1, 1)$, $\alpha = 1/2$, $\beta = 1/2$. Then

$$f(\alpha x + \beta y) = 0 \neq \alpha f(x) + \beta f(y) = 1.$$

Affine functions. A linear function plus a constant is called an *affine* function. A function $f : \mathbf{R}^n \to \mathbf{R}$ is affine if and only if it can be expressed as $f(x) = a^T x + b$ for some n-vector a and scalar b, which is sometimes called the *offset*. For example, the function on 3-vectors defined by

$$f(x) = 2.3 - 2x_1 + 1.3x_2 - x_3,$$

is affine, with $b = 2.3$, $a = (-2, 1.3, -1)$.

Any affine scalar-valued function satisfies the following variation on the superposition property:

$$f(\alpha x + \beta y) = \alpha f(x) + \beta f(y),$$

for all n-vectors x, y, and all scalars α, β that satisfy $\alpha + \beta = 1$. For linear functions, superposition holds for *any* coefficients α and β; for affine functions, it holds *when the coefficients sum to one* (*i.e.*, when the argument is an affine combination).

To see that the restricted superposition property holds for an affine function $f(x) = a^T x + b$, we note that, for any vectors x, y and scalars α and β that satisfy $\alpha + \beta = 1$,

$$\begin{aligned} f(\alpha x + \beta y) &= a^T(\alpha x + \beta y) + b \\ &= \alpha a^T x + \beta a^T y + (\alpha + \beta)b \\ &= \alpha(a^T x + b) + \beta(a^T y + b) \\ &= \alpha f(x) + \beta f(y). \end{aligned}$$

(In the second line we use $\alpha + \beta = 1$.)

This restricted superposition property for affine functions is useful in showing that a function f is *not* affine: We find vectors x, y, and numbers α and β with $\alpha + \beta = 1$, and verify that $f(\alpha x + \beta y) \neq \alpha f(x) + \beta f(y)$. This shows that f cannot be affine. As an example, we verified above that superposition does not hold for the maximum function (with $n > 1$); the coefficients in our counterexample are $\alpha = \beta = 1/2$, which sum to one, which allows us to conclude that the maximum function is not affine.

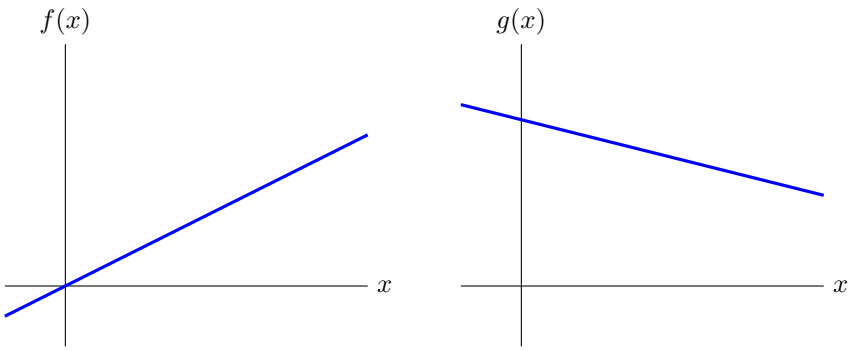

Figure 2.1 *Left.* The function f is linear. *Right.* The function g is affine, but not linear.

The converse is also true: Any scalar-valued function that satisfies the restricted superposition property is affine. An analog of the formula (2.3) is

$$f(x) = f(0) + x_1\left(f(e_1) - f(0)\right) + \cdots + x_n\left(f(e_n) - f(0)\right), \tag{2.4}$$

which holds when f is affine, and x is any n-vector. (See exercise 2.7.) This formula shows that for an affine function, once we know the $n+1$ numbers $f(0)$, $f(e_1)$, $\ldots$, $f(e_n)$, we can predict (or reconstruct or evaluate) $f(x)$ for any n-vector x. It also shows how the vector a and constant b in the representation $f(x) = a^T x + b$ can be found from the function f: $a_i = f(e_i) - f(0)$, and $b = f(0)$.

In some contexts affine functions are called linear. For example, when x is a scalar, the function f defined as $f(x) = \alpha x + \beta$ is sometimes referred to as a linear function of x, perhaps because its graph is a line. But when $\beta \neq 0$, f is not a linear function of x, in the standard mathematical sense; it *is* an affine function of x. In this book we will distinguish between linear and affine functions. Two simple examples are shown in figure 2.1.

A civil engineering example. Many scalar-valued functions that arise in science and engineering are well approximated by linear or affine functions. As a typical example, consider a steel structure like a bridge, and let w be an n-vector that gives the weight of the load on the bridge in n specific locations, in metric tons. These loads will cause the bridge to deform (move and change shape) slightly. Let s denote the distance that a specific point on the bridge sags, in millimeters, due to the load w. This is shown in figure 2.2. For weights the bridge is designed to handle, the sag is very well approximated as a linear function $s = f(x)$. This function can be expressed as an inner product, $s = c^T w$, for some n-vector c. From the equation $s = c_1 w_1 + \cdots + c_n w_n$, we see that $c_1 w_1$ is the amount of the sag that is due to the weight w_1, and similarly for the other weights. The coefficients c_i, which have units of mm/ton, are called *compliances*, and give the sensitivity of the sag with respect to loads applied at the n locations.

The vector c can be computed by (numerically) solving a partial differential equation, given the detailed design of the bridge and the mechanical properties of

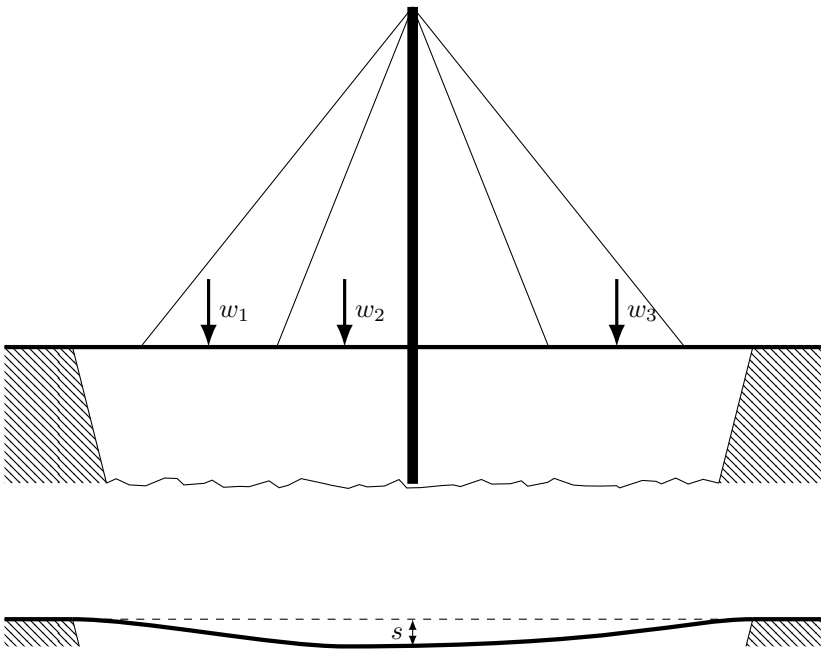

Figure 2.2 A bridge with weights w_1, w_2, w_3 applied in 3 locations. These weights cause the bridge to sag in the middle, by an amount s. (The sag is exaggerated in this diagram.)

w_1	w_2	w_3	Measured sag	Predicted sag
1	0	0	0.12	—
0	1	0	0.31	—
0	0	1	0.26	—
0.5	1.1	0.3	0.481	0.479
1.5	0.8	1.2	0.736	0.740

Table 2.1 Loadings on a bridge (first three columns), the associated measured sag at a certain point (fourth column), and the predicted sag using the linear model constructed from the first three experiments (fifth column).

the steel used to construct it. This is always done during the design of a bridge. The vector c can also be *measured* once the bridge is built, using the formula (2.3). We apply the load $w = e_1$, which means that we place a one ton load at the first load position on the bridge, with no load at the other positions. We can then measure the sag, which is c_1. We repeat this experiment, moving the one ton load to positions $2, 3, \ldots, n$, which gives us the coefficients $c_2, \ldots, c_n$. At this point we have the vector c, so we can now *predict* what the sag will be with any other loading. To check our measurements (and linearity of the sag function) we might measure the sag under other more complicated loadings, and in each case compare our prediction (*i.e.*, $c^T w$) with the actual measured sag.

Table 2.1 shows what the results of these experiments might look like, with each row representing an experiment (*i.e.*, placing the loads and measuring the sag). In the last two rows we compare the measured sag and the predicted sag, using the linear function with coefficients found in the first three experiments.

2.2 Taylor approximation

In many applications, scalar-valued functions of n variables, or relations between n variables and a scalar one, can be *approximated* as linear or affine functions. In these cases we sometimes refer to the linear or affine function relating the variables and the scalar variable as a *model*, to remind us that the relation is only an approximation, and not exact.

Differential calculus gives us an organized way to find an approximate affine model. Suppose that $f : \mathbf{R}^n \to \mathbf{R}$ is differentiable, which means that its partial derivatives exist (see §C.1). Let z be an n-vector. The (first-order) *Taylor approximation* of f near (or at) the point z is the function $\hat{f}(x)$ of x defined as

$$\hat{f}(x) = f(z) + \frac{\partial f}{\partial x_1}(z)(x_1 - z_1) + \cdots + \frac{\partial f}{\partial x_n}(z)(x_n - z_n),$$

where $\frac{\partial f}{\partial x_i}(z)$ denotes the partial derivative of f with respect to its ith argument, evaluated at the n-vector z. The hat appearing over f on the left-hand side is

a common notational hint that it is an approximation of the function f. (The approximation is named after the mathematician Brook Taylor.)

The first-order Taylor approximation $\hat{f}(x)$ is a very good approximation of $f(x)$ when all x_i are near the associated z_i. Sometimes $\hat{f}$ is written with a second vector argument, as $\hat{f}(x;z)$, to show the point z at which the approximation is developed. The first term in the Taylor approximation is a constant; the other terms can be interpreted as the contributions to the (approximate) change in the function value (from $f(z)$) due to the changes in the components of x (from z).

Evidently $\hat{f}$ is an affine function of x. (It is sometimes called the *linear approximation* of f near z, even though it is in general affine, and not linear.) It can be written compactly using inner product notation as

$$\hat{f}(x) = f(z) + \nabla f(z)^T(x - z), \tag{2.5}$$

where $\nabla f(z)$ is an n-vector, the *gradient of* f (at the point z),

$$\nabla f(z) = \begin{bmatrix} \frac{\partial f}{\partial x_1}(z) \\ \vdots \\ \frac{\partial f}{\partial x_n}(z) \end{bmatrix}. \tag{2.6}$$

The first term in the Taylor approximation (2.5) is the constant $f(z)$, the value of the function when $x = z$. The second term is the inner product of the gradient of f at z and the *deviation* or *perturbation* of x from z, *i.e.*, $x - z$.

We can express the first-order Taylor approximation as a linear function plus a constant,

$$\hat{f}(x) = \nabla f(z)^T x + (f(z) - \nabla f(z)^T z),$$

but the form (2.5) is perhaps easier to interpret.

The first-order Taylor approximation gives us an organized way to construct an affine approximation of a function $f : \mathbf{R}^n \to \mathbf{R}$, near a given point z, when there is a formula or equation that describes f, and it is differentiable. A simple example, for $n = 1$, is shown in figure 2.3. Over the full x-axis scale shown, the Taylor approximation $\hat{f}$ does not give a good approximation of the function f. But for x near z, the Taylor approximation is very good.

Example. Consider the function $f : \mathbf{R}^2 \to \mathbf{R}$ given by $f(x) = x_1 + \exp(x_2 - x_1)$, which is not linear or affine. To find the Taylor approximation $\hat{f}$ near the point $z = (1, 2)$, we take partial derivatives to obtain

$$\nabla f(z) = \begin{bmatrix} 1 - \exp(z_2 - z_1) \\ \exp(z_2 - z_1) \end{bmatrix},$$

which evaluates to $(-1.7183, 2.7183)$ at $z = (1, 2)$. The Taylor approximation at $z = (1, 2)$ is then

$$\begin{aligned} \hat{f}(x) &= 3.7183 + (-1.7183, 2.7183)^T(x - (1, 2)) \\ &= 3.7183 - 1.7183(x_1 - 1) + 2.7183(x_2 - 2). \end{aligned}$$

Table 2.2 shows $f(x)$ and $\hat{f}(x)$, and the approximation error $|\hat{f}(x) - f(x)|$, for some values of x relatively near z. We can see that $\hat{f}$ is indeed a very good approximation of f, especially when x is near z.

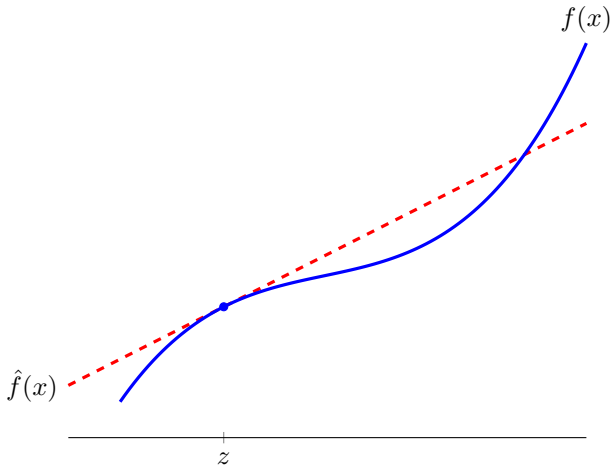

Figure 2.3 A function f of one variable, and the first-order Taylor approximation $\hat{f}(x) = f(z) + f'(z)(x - z)$ at z.

x	$f(x)$	$\hat{f}(x)$	$\lvert\hat{f}(x) - f(x)\rvert$
$(1.00, 2.00)$	3.7183	3.7183	0.0000
$(0.96, 1.98)$	3.7332	3.7326	0.0005
$(1.10, 2.11)$	3.8456	3.8455	0.0001
$(0.85, 2.05)$	4.1701	4.1119	0.0582
$(1.25, 2.41)$	4.4399	4.4032	0.0367

Table 2.2 Some values of x (first column), the function value $f(x)$ (second column), the Taylor approximation $\hat{f}(x)$ (third column), and the error (fourth column).

2.3 Regression model

In this section we describe a very commonly used affine function, especially when the n-vector x represents a feature vector. The affine function of x given by

$$\hat{y} = x^T\beta + v, \tag{2.7}$$

where β is an n-vector and v is a scalar, is called a *regression model*. In this context, the entries of x are called the *regressors*, and $\hat{y}$ is called the *prediction*, since the regression model is typically an approximation or prediction of some true value y, which is called the *dependent variable*, *outcome*, or *label*.

The vector β is called the *weight vector* or *coefficient vector*, and the scalar v is called the *offset* or *intercept* in the regression model. Together, β and v are called the *parameters* in the regression model. (We will see in chapter 13 how the parameters in a regression model can be estimated or guessed, based on some past or known observations of the feature vector x and the associated outcome y.) The symbol $\hat{y}$ is used in the regression model to emphasize that it is an *estimate* or *prediction* of some outcome y.

The entries in the weight vector have a simple interpretation: β_i is the amount by which $\hat{y}$ increases (if $\beta_i > 0$) when feature i increases by one (with all other features the same). If β_i is small, the prediction $\hat{y}$ doesn't depend too strongly on feature i. The offset v is the value of $\hat{y}$ when all features have the value 0.

The regression model is very interpretable when all of the features are Boolean, *i.e.*, have values that are either 0 or 1, which occurs when the features represent which of two outcomes holds. As a simple example consider a regression model for the lifespan of a person in some group, with $x_1 = 0$ if the person is female ($x_1 = 1$ if male), $x_2 = 1$ if the person has type II diabetes, and $x_3 = 1$ if the person smokes cigarettes. In this case, v is the regression model estimate for the lifespan of a female nondiabetic nonsmoker; β_1 is the increase in estimated lifespan if the person is male, β_2 is the increase in estimated lifespan if the person is diabetic, and β_3 is the increase in estimated lifespan if the person smokes cigarettes. (In a model that fits real data, all three of these coefficients would be negative, meaning that they decrease the regression model estimate of lifespan.)

Simplified regression model notation. Vector stacking can be used to lump the weights and offset in the regression model (2.7) into a single parameter vector, which simplifies the regression model notation a bit. We create a new regressor vector $\tilde{x}$, with $n + 1$ entries, as $\tilde{x} = (1, x)$. We can think of $\tilde{x}$ as a new feature vector, consisting of all n original features, and one new feature added ($\tilde{x}_1$) at the beginning, which always has the value one. We define the parameter vector $\tilde{\beta} = (v, \beta)$, so the regression model (2.7) has the simple inner product form

$$\hat{y} = x^T\beta + v = \begin{bmatrix} 1 \\ x \end{bmatrix}^T \begin{bmatrix} v \\ \beta \end{bmatrix} = \tilde{x}^T\tilde{\beta}. \tag{2.8}$$

Often we omit the tildes, and simply write this as $\hat{y} = x^T\beta$, where we assume that the first feature in x is the constant 1. A feature that always has the value 1 is not particularly informative or interesting, but it does simplify the notation in a regression model.

House	x_1 (area)	x_2 (beds)	y (price)	$\hat{y}$ (prediction)
1	0.846	1	115.00	161.37
2	1.324	2	234.50	213.61
3	1.150	3	198.00	168.88
4	3.037	4	528.00	430.67
5	3.984	5	572.50	552.66

Table 2.3 Five houses with associated feature vectors shown in the second and third columns. The fourth and fifth column give the actual price, and the price predicted by the regression model.

House price regression model. As a simple example of a regression model, suppose that y is the selling price of a house in some neighborhood, over some time period, and the 2-vector x contains attributes of the house:

- x_1 is the house area (in 1000 square feet),
- x_2 is the number of bedrooms.

If y represents the selling price of the house, in thousands of dollars, the regression model

$$\hat{y} = x^T\beta + v = \beta_1 x_1 + \beta_2 x_2 + v$$

predicts the price in terms of the attributes or features. This regression model is not meant to describe an exact relationship between the house attributes and its selling price; it is a model or approximation. Indeed, we would expect such a model to give, at best, only a crude approximation of selling price.

As a specific numerical example, consider the regression model parameters

$$\beta = (148.73, -18.85), \qquad v = 54.40. \tag{2.9}$$

These parameter values were found using the methods we will see in chapter 13, based on records of sales for 774 houses in the Sacramento area. Table 2.3 shows the feature vectors x for five houses that sold during the period, the actual sale price y, and the predicted price $\hat{y}$ from the regression model above. Figure 2.4 shows the predicted and actual sale prices for 774 houses, including the five houses in the table, on a scatter plot, with actual price on the horizontal axis and predicted price on the vertical axis.

We can see that this particular regression model gives reasonable, but not very accurate, predictions of the actual sale price. (Regression models for house prices that are used in practice use many more than two regressors, and are much more accurate.)

The model parameters in (2.9) are readily interpreted. The parameter $\beta_1 = 148.73$ is the amount the regression model price prediction increases (in thousands of dollars) when the house area increases by 1000 square feet (with the same number of bedrooms). The parameter $\beta_2 = -18.85$ is the price prediction increase with the addition of one bedroom, with the total house area held constant, in units of

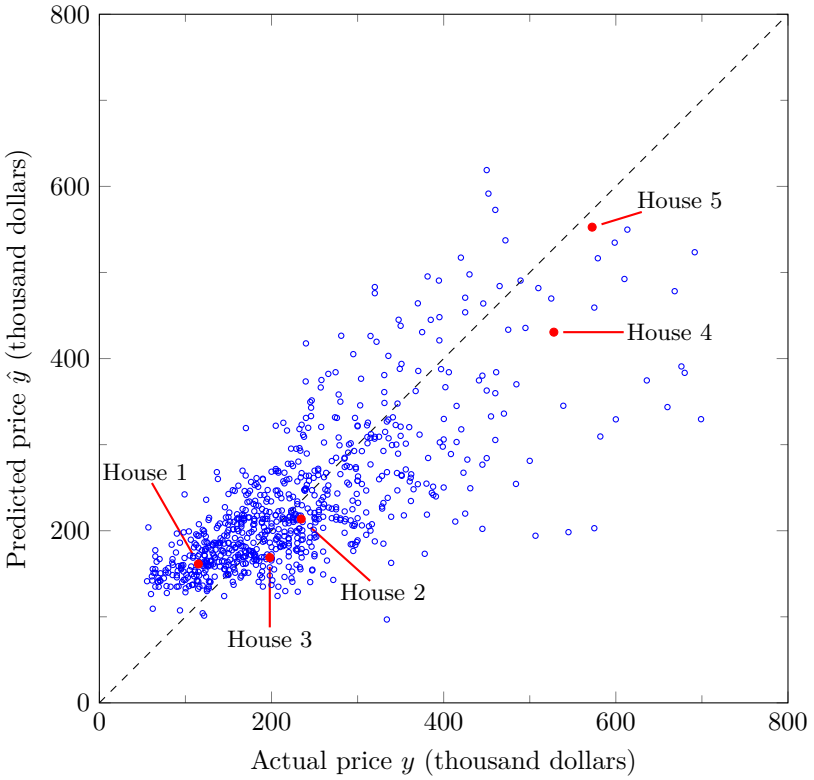

Figure 2.4 Scatter plot of actual and predicted sale prices for 774 houses sold in Sacramento during a five-day period.

thousands of dollars per bedroom. It might seem strange that β_2 is negative, since one imagines that adding a bedroom to a house would *increase* its sale price, not decrease it. To understand why β_2 might be negative, we note that it gives the change in predicted price when we add a bedroom, without adding any additional area to the house. If we remodel a house by adding a bedroom that *also* adds more than around 127 square feet to the house area, the regression model (2.9) *does* predict an increase in house sale price. The offset $v = 54.40$ is the predicted price for a house with no area and no bedrooms, which we might interpret as the model's prediction of the value of the lot. But this regression model is crude enough that these interpretations are dubious.

Exercises

2.1 *Linear or not?* Determine whether each of the following scalar-valued functions of n-vectors is linear. If it is a linear function, give its inner product representation, *i.e.*, an n-vector a for which $f(x) = a^T x$ for all x. If it is not linear, give specific x, y, α, and β for which superposition fails, *i.e.*,

$$f(\alpha x + \beta y) \neq \alpha f(x) + \beta f(y).$$

(a) The spread of values of the vector, defined as $f(x) = \max_k x_k - \min_k x_k$.

(b) The difference of the last element and the first, $f(x) = x_n - x_1$.

(c) The median of an n-vector, where we will assume $n = 2k + 1$ is odd. The median of the vector x is defined as the $(k + 1)$st largest number among the entries of x. For example, the median of $(-7.1, 3.2, -1.5)$ is -1.5.

(d) The average of the entries with odd indices, minus the average of the entries with even indices. You can assume that $n = 2k$ is even.

(e) Vector extrapolation, defined as $x_n + (x_n - x_{n-1})$, for $n \geq 2$. (This is a simple prediction of what x_{n+1} would be, based on a straight line drawn through x_n and x_{n-1}.)

2.2 *Processor powers and temperature.* The temperature T of an electronic device containing three processors is an affine function of the power dissipated by the three processors, $P = (P_1, P_2, P_3)$. When all three processors are idling, we have $P = (10, 10, 10)$, which results in a temperature $T = 30$. When the first processor operates at full power and the other two are idling, we have $P = (100, 10, 10)$, and the temperature rises to $T = 60$. When the second processor operates at full power and the other two are idling, we have $P = (10, 100, 10)$ and $T = 70$. When the third processor operates at full power and the other two are idling, we have $P = (10, 10, 100)$ and $T = 65$. Now suppose that all three processors are operated at the same power P^{same}. How large can P^{same} be, if we require that $T \leq 85$? *Hint.* From the given data, find the 3-vector a and number b for which $T = a^T P + b$.

2.3 *Motion of a mass in response to applied force.* A unit mass moves on a straight line (in one dimension). The position of the mass at time t (in seconds) is denoted by $s(t)$, and its derivatives (the velocity and acceleration) by $s'(t)$ and $s''(t)$. The position as a function of time can be determined from Newton's second law

$$s''(t) = F(t),$$

where $F(t)$ is the force applied at time t, and the initial conditions $s(0)$, $s'(0)$. We assume $F(t)$ is piecewise-constant, and is kept constant in intervals of one second. The sequence of forces $F(t)$, for $0 \leq t < 10$, can then be represented by a 10-vector f, with

$$F(t) = f_k, \quad k - 1 \leq t < k.$$

Derive expressions for the final velocity $s'(10)$ and final position $s(10)$. Show that $s(10)$ and $s'(10)$ are affine functions of x, and give 10-vectors a, c and constants b, d for which

$$s'(10) = a^T f + b, \qquad s(10) = c^T f + d.$$

This means that the mapping from the applied force sequence to the final position and velocity is affine.

Hint. You can use

$$s'(t) = s'(0) + \int_0^t F(\tau)\, d\tau, \qquad s(t) = s(0) + \int_0^t s'(\tau)\, d\tau.$$

You will find that the mass velocity $s'(t)$ is piecewise-linear.

2.4 *Linear function?* The function $\phi : \mathbf{R}^3 \to \mathbf{R}$ satisfies

$$\phi(1,1,0) = -1, \qquad \phi(-1,1,1) = 1, \qquad \phi(1,-1,-1) = 1.$$

Choose one of the following, and justify your choice: ϕ must be linear; ϕ could be linear; ϕ cannot be linear.

2.5 *Affine function.* Suppose $\psi : \mathbf{R}^2 \to \mathbf{R}$ is an affine function, with $\psi(1,0) = 1$, $\psi(1,-2) = 2$.

(a) What can you say about $\psi(1,-1)$? Either give the value of $\psi(1,-1)$, or state that it cannot be determined.

(b) What can you say about $\psi(2,-2)$? Either give the value of $\psi(2,-2)$, or state that it cannot be determined.

Justify your answers.

2.6 *Questionnaire scoring.* A questionnaire in a magazine has 30 questions, broken into two sets of 15 questions. Someone taking the questionnaire answers each question with 'Rarely', 'Sometimes', or 'Often'. The answers are recorded as a 30-vector a, with $a_i = 1, 2, 3$ if question i is answered Rarely, Sometimes, or Often, respectively. The total score on a completed questionnaire is found by adding up 1 point for every question answered Sometimes and 2 points for every question answered Often on questions 1–15, and by adding 2 points and 4 points for those responses on questions 16–30. (Nothing is added to the score for Rarely responses.) Express the total score s in the form of an affine function $s = w^T a + v$, where w is a 30-vector and v is a scalar (number).

2.7 *General formula for affine functions.* Verify that formula (2.4) holds for any affine function $f : \mathbf{R}^n \to \mathbf{R}$. You can use the fact that $f(x) = a^T x + b$ for some n-vector a and scalar b.

2.8 *Integral and derivative of polynomial.* Suppose the n-vector c gives the coefficients of a polynomial $p(x) = c_1 + c_2 x + \cdots + c_n x^{n-1}$.

(a) Let α and β be numbers with $\alpha < \beta$. Find an n-vector a for which

$$a^T c = \int_\alpha^\beta p(x)\, dx$$

always holds. This means that the integral of a polynomial over an interval is a linear function of its coefficients.

(b) Let α be a number. Find an n-vector b for which

$$b^T c = p'(\alpha).$$

This means that the derivative of the polynomial at a given point is a linear function of its coefficients.

2.9 *Taylor approximation.* Consider the function $f : \mathbf{R}^2 \to \mathbf{R}$ given by $f(x_1, x_2) = x_1 x_2$. Find the Taylor approximation $\hat{f}$ at the point $z = (1,1)$. Compare $f(x)$ and $\hat{f}(x)$ for the following values of x:

$$x = (1,1), \quad x = (1.05, 0.95), \quad x = (0.85, 1.25), \quad x = (-1, 2).$$

Make a brief comment about the accuracy of the Taylor approximation in each case.

2.10 *Regression model.* Consider the regression model $\hat{y} = x^T \beta + v$, where $\hat{y}$ is the predicted response, x is an 8-vector of features, β is an 8-vector of coefficients, and v is the offset term. Determine whether each of the following statements is true or false.

(a) If $\beta_3 > 0$ and $x_3 > 0$, then $\hat{y} \geq 0$.

(b) If $\beta_2 = 0$ then the prediction $\hat{y}$ does not depend on the second feature x_2.

(c) If $\beta_6 = -0.8$, then increasing x_6 (keeping all other x_is the same) will decrease $\hat{y}$.

2.11 *Sparse regression weight vector.* Suppose that x is an n-vector that gives n features for some object, and the scalar y is some outcome associated with the object. What does it mean if a regression model $\hat{y} = x^T\beta + v$ uses a sparse weight vector β? Give your answer in English, referring to $\hat{y}$ as our prediction of the outcome.

2.12 *Price change to maximize profit.* A business sells n products, and is considering changing the price of *one* of the products to increase its total profits. A business analyst develops a regression model that (reasonably accurately) predicts the total profit when the product prices are changed, given by $\hat{P} = \beta^T x + P$, where the n-vector x denotes the fractional change in the product prices, $x_i = (p_i^{\text{new}} - p_i)/p_i$. Here P is the profit with the current prices, $\hat{P}$ is the predicted profit with the changed prices, p_i is the current (positive) price of product i, and p_i^{new} is the new price of product i.

(a) What does it mean if $\beta_3 < 0$? (And yes, this can occur.)

(b) Suppose that you are given permission to change the price of *one* product, by up to 1%, to increase total profit. Which product would you choose, and would you increase or decrease the price? By how much?

(c) Repeat part (b) assuming you are allowed to change the price of two products, each by up to 1%.

Chapter 3

Norm and distance

In this chapter we focus on the norm of a vector, a measure of its magnitude, and on related concepts like distance, angle, standard deviation, and correlation.

3.1 Norm

The *Euclidean norm* of an n-vector x (named after the Greek mathematician Euclid), denoted $\|x\|$, is the squareroot of the sum of the squares of its elements,

$$\|x\| = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}.$$

The Euclidean norm can also be expressed as the squareroot of the inner product of the vector with itself, *i.e.*, $\|x\| = \sqrt{x^T x}$.

The Euclidean norm is sometimes written with a subscript 2, as $\|x\|_2$. (The subscript 2 indicates that the entries of x are raised to the second power.) Other less widely used terms for the Euclidean norm of a vector are the *magnitude*, or *length*, of a vector. (The term *length* should be avoided, since it is also often used to refer to the dimension of the vector.) We use the same notation for the norms of vectors of different dimensions.

As simple examples, we have

$$\left\| \begin{bmatrix} 2 \\ -1 \\ 2 \end{bmatrix} \right\| = \sqrt{9} = 3, \qquad \left\| \begin{bmatrix} 0 \\ -1 \end{bmatrix} \right\| = 1.$$

When x is a scalar, *i.e.*, a 1-vector, the Euclidean norm is the same as the absolute value of x. Indeed, the Euclidean norm can be considered a generalization or extension of the absolute value or magnitude, that applies to vectors. The double bar notation is meant to suggest this. Like the absolute value of a number, the norm of a vector is a (numerical) measure of its magnitude. We say a vector is *small* if its norm is a small number, and we say it is *large* if its norm is a large number. (The numerical values of the norm that qualify for small or large depend on the particular application and context.)

Properties of norm. Some important properties of the Euclidean norm are given below. Here x and y are vectors of the same size, and β is a scalar.

- *Nonnegative homogeneity.* $\|\beta x\| = |\beta|\|x\|$. Multiplying a vector by a scalar multiplies the norm by the absolute value of the scalar.
- *Triangle inequality.* $\|x+y\| \leq \|x\|+\|y\|$. The Euclidean norm of a sum of two vectors is no more than the sum of their norms. (The name of this property will be explained later.) Another name for this inequality is *subadditivity*.
- *Nonnegativity.* $\|x\| \geq 0$.
- *Definiteness.* $\|x\| = 0$ only if $x = 0$.

The last two properties together, which state that the norm is always nonnegative, and zero only when the vector is zero, are called *positive definiteness*. The first, third, and fourth properties are easy to show directly from the definition of the norm. As an example, let's verify the definiteness property. If $\|x\| = 0$, then we also have $\|x\|^2 = 0$, which means that $x_1^2 + \cdots + x_n^2 = 0$. This is a sum of n nonnegative numbers, which is zero. We can conclude that each of the n numbers is zero, since if any of them were nonzero the sum would be positive. So we conclude that $x_i^2 = 0$ for $i = 1, \ldots, n$, and therefore $x_i = 0$ for $i = 1, \ldots, n$; and thus, $x = 0$. Establishing the second property, the triangle inequality, is not as easy; we will give a derivation on page 57.

General norms. Any real-valued function of an n-vector that satisfies the four properties listed above is called a (general) norm. But in this book we will only use the Euclidean norm, so from now on, we refer to the Euclidean norm as the norm. (See exercise 3.5, which describes some other useful norms.)

Root-mean-square value. The norm is related to the *root-mean-square* (RMS) value of an n-vector x, defined as

$$\mathbf{rms}(x) = \sqrt{\frac{x_1^2 + \cdots + x_n^2}{n}} = \frac{\|x\|}{\sqrt{n}}.$$

The argument of the squareroot in the middle expression is called the *mean square* value of x, denoted $\mathbf{ms}(x)$, and the RMS value is the squareroot of the mean square value. The RMS value of a vector x is useful when comparing norms of vectors with different dimensions; the RMS value tells us what a 'typical' value of $|x_i|$ is. For example, the norm of $\mathbf{1}$, the n-vector of all ones, is $\sqrt{n}$, but its RMS value is 1, independent of n. More generally, if all the entries of a vector are the same, say, α, then the RMS value of the vector is $|\alpha|$.

Norm of a sum. A useful formula for the norm of the sum of two vectors x and y is

$$\|x + y\| = \sqrt{\|x\|^2 + 2x^T y + \|y\|^2}. \tag{3.1}$$

To derive this formula, we start with the square of the norm of $x+y$ and use various properties of the inner product:

$$\begin{aligned}\|x+y\|^2 &= (x+y)^T(x+y)\\ &= x^Tx + x^Ty + y^Tx + y^Ty\\ &= \|x\|^2 + 2x^Ty + \|y\|^2.\end{aligned}$$

Taking the squareroot of both sides yields the formula (3.1) above. In the first line, we use the definition of the norm. In the second line, we expand the inner product. In the fourth line we use the definition of the norm, and the fact that $x^Ty = y^Tx$. Some other identities relating norms, sums, and inner products of vectors are explored in exercise 3.4.

Norm of block vectors. The norm-squared of a stacked vector is the sum of the norm-squared values of its subvectors. For example, with $d = (a, b, c)$ (where a, b, and c are vectors), we have

$$\|d\|^2 = d^Td = a^Ta + b^Tb + c^Tc = \|a\|^2 + \|b\|^2 + \|c\|^2.$$

This idea is often used in reverse, to express the sum of the norm-squared values of some vectors as the norm-square value of a block vector formed from them.

We can write the equality above in terms of norms as

$$\|(a, b, c)\| = \sqrt{\|a\|^2 + \|b\|^2 + \|c\|^2} = \|(\|a\|, \|b\|, \|c\|)\|.$$

In words: The norm of a stacked vector is the norm of the vector formed from the norms of the subvectors. The right-hand side of the equation above should be carefully read. The outer norm symbols enclose a 3-vector, with (scalar) entries $\|a\|$, $\|b\|$, and $\|c\|$.

Chebyshev inequality. Suppose that x is an n-vector, and that k of its entries satisfy $|x_i| \geq a$, where $a > 0$. Then k of its entries satisfy $x_i^2 \geq a^2$. It follows that

$$\|x\|^2 = x_1^2 + \cdots + x_n^2 \geq ka^2,$$

since k of the numbers in the sum are at least a^2, and the other $n-k$ numbers are nonnegative. We can conclude that $k \leq \|x\|^2/a^2$, which is called the *Chebyshev inequality*, after the mathematician Pafnuty Chebyshev. When $\|x\|^2/a^2 \geq n$, the inequality tells us nothing, since we always have $k \leq n$. In other cases it limits the number of entries in a vector that can be large. For $a > \|x\|$, the inequality is $k \leq \|x\|^2/a^2 < 1$, so we conclude that $k = 0$ (since k is an integer). In other words, no entry of a vector can be larger in magnitude than the norm of the vector.

The Chebyshev inequality is easier to interpret in terms of the RMS value of a vector. We can write it as

$$\frac{k}{n} \leq \left(\frac{\mathbf{rms}(x)}{a}\right)^2, \tag{3.2}$$

where k is, as above, the number of entries of x with absolute value at least a. The left-hand side is the fraction of entries of the vector that are at least a in absolute

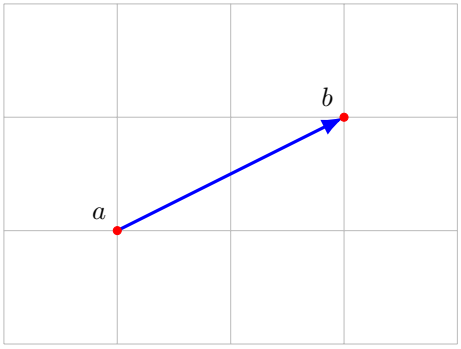

Figure 3.1 The norm of the displacement $b - a$ is the distance between the points with coordinates a and b.

value. The right-hand side is the inverse square of the ratio of a to $\mathbf{rms}(x)$. It says, for example, that no more than $1/25 = 4\%$ of the entries of a vector can exceed its RMS value by more than a factor of 5. The Chebyshev inequality partially justifies the idea that the RMS value of a vector gives an idea of the size of a typical entry: It states that not too many of the entries of a vector can be much bigger (in absolute value) than its RMS value. (A converse statement can also be made: At least one entry of a vector has absolute value as large as the RMS value of the vector; see exercise 3.8.)

3.2 Distance

Euclidean distance. We can use the norm to define the *Euclidean distance* between two vectors a and b as the norm of their difference:

$$\mathbf{dist}(a, b) = \|a - b\|.$$

For one, two, and three dimensions, this distance is exactly the usual distance between points with coordinates a and b, as illustrated in figure 3.1. But the Euclidean distance is defined for vectors of any dimension; we can refer to the distance between two vectors of dimension 100. Since we only use the Euclidean norm in this book, we will refer to the Euclidean distance between vectors as, simply, the distance between the vectors. If a and b are n-vectors, we refer to the RMS value of the difference, $\|a - b\|/\sqrt{n}$, as the *RMS deviation* between the two vectors.

When the distance between two n-vectors x and y is small, we say they are 'close' or 'nearby', and when the distance $\|x - y\|$ is large, we say they are 'far'. The particular numerical values of $\|x-y\|$ that correspond to 'close' or 'far' depend on the particular application.

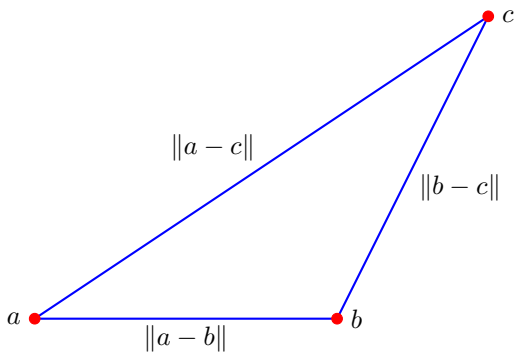

Figure 3.2 Triangle inequality.

As an example, consider the 4-vectors

$$u = \begin{bmatrix} 1.8 \\ 2.0 \\ -3.7 \\ 4.7 \end{bmatrix}, \qquad v = \begin{bmatrix} 0.6 \\ 2.1 \\ 1.9 \\ -1.4 \end{bmatrix}, \qquad w = \begin{bmatrix} 2.0 \\ 1.9 \\ -4.0 \\ 4.6 \end{bmatrix}.$$

The distances between pairs of them are

$$\|u - v\| = 8.368, \qquad \|u - w\| = 0.387, \qquad \|v - w\| = 8.533,$$

so we can say that u is much nearer (or closer) to w than it is to v. We can also say that w is much nearer to u than it is to v.

Triangle inequality. We can now explain where the triangle inequality gets its name. Consider a triangle in two or three dimensions, whose vertices have coordinates a, b, and c. The lengths of the sides are the distances between the vertices,

$$\mathbf{dist}(a, b) = \|a - b\|, \qquad \mathbf{dist}(b, c) = \|b - c\|, \qquad \mathbf{dist}(a, c) = \|a - c\|.$$

Geometric intuition tells us that the length of any side of a triangle cannot exceed the sum of the lengths of the other two sides. For example, we have

$$\|a - c\| \leq \|a - b\| + \|b - c\|. \tag{3.3}$$

This follows from the triangle inequality, since

$$\|a - c\| = \|(a - b) + (b - c)\| \leq \|a - b\| + \|b - c\|.$$

This is illustrated in figure 3.2.

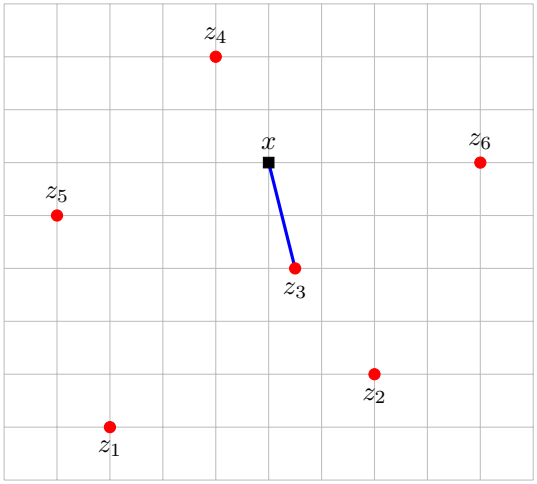

Figure 3.3 A point x, shown as a square, and six other points $z_1, \ldots, z_6$. The point z_3 is the nearest neighbor of x among the points $z_1, \ldots, z_6$.

Examples.

- *Feature distance.* If x and y represent vectors of n features of two objects, the quantity $\|x - y\|$ is called the *feature distance*, and gives a measure of how different the objects are (in terms of their feature values). Suppose for example the feature vectors are associated with patients in a hospital, with entries such as weight, age, presence of chest pain, difficulty breathing, and the results of tests. We can use feature vector distance to say that one patient case is near another one (at least in terms of their feature vectors).

- *RMS prediction error.* Suppose that the n-vector y represents a time series of some quantity, for example, hourly temperature at some location, and $\hat{y}$ is another n-vector that represents an estimate or prediction of the time series y, based on other information. The difference $y - \hat{y}$ is called the *prediction error*, and its RMS value $\mathbf{rms}(y - \hat{y})$ is called the *RMS prediction error*. If this value is small (say, compared to $\mathbf{rms}(y)$) the prediction is good.

- *Nearest neighbor.* Suppose $z_1, \ldots, z_m$ is a collection of m n-vectors, and that x is another n-vector. We say that z_j is the *nearest neighbor* of x (among $z_1, \ldots, z_m$) if
$$\|x - z_j\| \leq \|x - z_i\|, \quad i = 1, \ldots, m.$$
In words: z_j is the closest vector to x among the vectors $z_1, \ldots, z_m$. This is illustrated in figure 3.3. The idea of nearest neighbor, and generalizations such as the k-nearest neighbors, are used in many applications.

- *Document dissimilarity.* Suppose n-vectors x and y represent the histograms of word occurrences for two documents. Then $\|x - y\|$ represents a measure of the dissimilarity of the two documents. We might expect the dissimilarity

	Veterans Day	Memorial Day	Academy Awards	Golden Globe Awards	Super Bowl
Veterans Day	0	0.095	0.130	0.153	0.170
Memorial Day	0.095	0	0.122	0.147	0.164
Academy A.	0.130	0.122	0	0.108	0.164
Golden Globe A.	0.153	0.147	0.108	0	0.181
Super Bowl	0.170	0.164	0.164	0.181	0

Table 3.1 Pairwise word count histogram distances between five Wikipedia articles.

to be smaller when the two documents have the same genre, topic, or author; we would expect it to be larger when they are on different topics, or have different authors. As an example we form the word count histograms for the 5 Wikipedia articles with titles 'Veterans Day', 'Memorial Day', 'Academy Awards', 'Golden Globe Awards', and 'Super Bowl', using a dictionary of 4423 words. (More detail is given in §4.4.) The pairwise distances between the word count histograms are shown in table 3.1. We can see that pairs of related articles have smaller word count histogram distances than less related pairs of articles.

Units for heterogeneous vector entries. The square of the distance between two n-vectors x and y is given by

$$\|x - y\|^2 = (x_1 - y_1)^2 + \cdots + (x_n - y_n)^2,$$

the sum of the squares of the differences between their respective entries. Roughly speaking, the entries in the vectors all have equal status in determining the distance between them. For example, if x_2 and y_2 differ by one, the contribution to the square of the distance between them is the same as the contribution when x_3 and y_3 differ by one. This makes sense when the entries of the vectors x and y represent the same type of quantity, using the same units (say, at different times or locations), for example meters or dollars. For example if x and y are word count histograms, their entries are all word occurrence frequencies, and it makes sense to say they are close when their distance is small.

When the entries of a vector represent different types of quantities, for example when the vector entries represent different types of features associated with an object, we must be careful about choosing the units used to represent the numerical values of the entries. If we want the different entries to have approximately equal status in determining distance, their numerical values should be approximately of the same magnitude. For this reason units for different entries in vectors are often chosen in such a way that their typical numerical values are similar in magnitude, so that the different entries play similar roles in determining distance.

As an example suppose that the 2-vectors x, y, and z are the feature vectors for three houses that were sold, as in the example described on page 39. The first entry of each vector gives the house area and the second entry gives the number of

bedrooms. These are very different types of features, since the first one is a physical area, and the second one is a count, *i.e.*, an integer. In the example on page 39, we chose the unit used to represent the first feature, area, to be thousands of square feet. With this choice of unit used to represent house area, the numerical values of both of these features range from around 1 to 5; their values have roughly the same magnitude. When we determine the distance between feature vectors associated with two houses, the difference in the area (in thousands of square feet), and the difference in the number of bedrooms, play equal roles.

For example, consider three houses with feature vectors

$$x = (1.6, 2), \quad y = (1.5, 2), \quad z = (1.6, 4).$$

The first two are 'close' or 'similar' since $\|x - y\| = 0.1$ is small (compared to the norms of x and y, which are around 2.5). This matches our intuition that the first two houses are similar, since they both have two bedrooms and are close in area. The third house would be considered 'far' or 'different' from the first two houses, and rightly so since it has four bedrooms instead of two.

To appreciate the significance of our choice of units in this example, suppose we had chosen instead to represent house area directly in square feet, and not thousands of square feet. The three houses above would then be represented by feature vectors

$$\tilde{x} = (1600, 2), \quad \tilde{y} = (1500, 2), \quad \tilde{z} = (1600, 4).$$

The distance between the first and third houses is now 2, which is very small compared to the norms of the vectors (which are around 1600). The distance between the first and second houses is much larger. It seems strange to consider a two-bedroom house and a four-bedroom house as 'very close', while two houses with the same number of bedrooms and similar areas are much more dissimilar. The reason is simple: With our choice of square feet as the unit to measure house area, distances are very strongly influenced by differences in area, with number of bedrooms playing a much smaller (relative) role.

3.3 Standard deviation

For any vector x, the vector $\tilde{x} = x - \mathbf{avg}(x)\mathbf{1}$ is called the associated *de-meaned* vector, obtained by subtracting from each entry of x the mean value of the entries. (This is not standard notation; *i.e.*, $\tilde{x}$ is not generally used to denote the de-meaned vector.) The mean value of the entries of $\tilde{x}$ is zero, *i.e.*, $\mathbf{avg}(\tilde{x}) = 0$. This explains why $\tilde{x}$ is called the de-meaned version of x; it is x with its mean removed. The de-meaned vector is useful for understanding how the entries of a vector deviate from their mean value. It is zero if all the entries in the original vector x are the same.

The *standard deviation* of an n-vector x is defined as the RMS value of the de-meaned vector $x - \mathbf{avg}(x)\mathbf{1}$, *i.e.*,

$$\mathbf{std}(x) = \sqrt{\frac{(x_1 - \mathbf{avg}(x))^2 + \cdots + (x_n - \mathbf{avg}(x))^2}{n}}.$$

This is the same as the RMS deviation between a vector x and the vector all of whose entries are $\mathbf{avg}(x)$. It can be written using the inner product and norm as

$$\mathbf{std}(x) = \frac{\|x - (\mathbf{1}^T x/n)\mathbf{1}\|}{\sqrt{n}}. \tag{3.4}$$

The standard deviation of a vector x tells us the typical amount by which its entries deviate from their average value. The standard deviation of a vector is zero only when all its entries are equal. The standard deviation of a vector is small when the entries of the vector are nearly the same.

As a simple example consider the vector $x = (1, -2, 3, 2)$. Its mean or average value is $\mathbf{avg}(x) = 1$, so the de-meaned vector is $\tilde{x} = (0, -3, 2, 1)$. Its standard deviation is $\mathbf{std}(x) = 1.872$. We interpret this number as a 'typical' value by which the entries differ from the mean of the entries. These numbers are 0, 3, 2, and 1, so 1.872 is reasonable.

We should warn the reader that another slightly different definition of the standard deviation of a vector is widely used, in which the denominator $\sqrt{n}$ in (3.4) is replaced with $\sqrt{n-1}$ (for $n \geq 2$). In this book we will only use the definition (3.4).

In some applications the Greek letter σ (sigma) is traditionally used to denote standard deviation, while the mean is denoted μ (mu). In this notation we have, for an n-vector x,

$$\mu = \mathbf{1}^T x/n, \qquad \sigma = \|x - \mu\mathbf{1}\|/\sqrt{n}.$$

We will use the symbols $\mathbf{avg}(x)$ and $\mathbf{std}(x)$, switching to μ and σ only with explanation, when describing an application that traditionally uses these symbols.

Average, RMS value, and standard deviation. The average, RMS value, and standard deviation of a vector are related by the formula

$$\mathbf{rms}(x)^2 = \mathbf{avg}(x)^2 + \mathbf{std}(x)^2. \tag{3.5}$$

This formula makes sense: $\mathbf{rms}(x)^2$ is the mean square value of the entries of x, which can be expressed as the square of the mean value, plus the mean square fluctuation of the entries of x around their mean value. We can derive this formula from our vector notation formula for $\mathbf{std}(x)$ given above. We have

$$\begin{aligned}
\mathbf{std}(x)^2 &= (1/n)\|x - (\mathbf{1}^T x/n)\mathbf{1}\|^2 \\
&= (1/n)(x^T x - 2x^T(\mathbf{1}^T x/n)\mathbf{1} + ((\mathbf{1}^T x/n)\mathbf{1})^T((\mathbf{1}^T x/n)\mathbf{1})) \\
&= (1/n)(x^T x - (2/n)(\mathbf{1}^T x)^2 + n(\mathbf{1}^T x/n)^2) \\
&= (1/n)x^T x - (\mathbf{1}^T x/n)^2 \\
&= \mathbf{rms}(x)^2 - \mathbf{avg}(x)^2,
\end{aligned}$$

which can be re-arranged to obtain the identity (3.5) above. This derivation uses many of the properties for norms and inner products, and should be read carefully to understand every step. In the second line, we expand the norm-square of the sum of two vectors. In the third line, we use the commutative property of scalar-vector multiplication, moving scalars such as $(\mathbf{1}^T x/n)$ to the front of each term, and also the fact that $\mathbf{1}^T\mathbf{1} = n$.

Examples.

- *Mean return and risk.* Suppose that an n-vector represents a time series of return on an investment, expressed as a percentage, in n time periods over some interval of time. Its average gives the mean return over the whole interval, often shortened to its *return*. Its standard deviation is a measure of how variable the return is, from period to period, over the time interval, *i.e.*, how much it typically varies from its mean, and is often called the (per period) *risk* of the investment. Multiple investments can be compared by plotting them on a *risk-return plot*, which gives the mean and standard deviation of the returns of each of the investments over some interval. A desirable return history vector has high mean return and low risk; this means that the returns in the different periods are consistently high. Figure 3.4 shows an example.

- *Temperature or rainfall.* Suppose that an n-vector is a time series of the daily average temperature at a particular location, over a one year period. Its average gives the average temperature at that location (over the year) and its standard deviation is a measure of how much the temperature varied from its average value. We would expect the average temperature to be high and the standard deviation to be low in a tropical location, and the opposite for a location with high latitude.

Chebyshev inequality for standard deviation. The Chebyshev inequality (3.2) can be transcribed to an inequality expressed in terms of the mean and standard deviation: If k is the number of entries of x that satisfy $|x_i - \mathbf{avg}(x)| \geq a$, then $k/n \leq (\mathbf{std}(x)/a)^2$. (This inequality is only interesting for $a > \mathbf{std}(x)$.) For example, at most $1/9 = 11.1\%$ of the entries of a vector can deviate from the mean value $\mathbf{avg}(x)$ by 3 standard deviations or more. Another way to state this is: The fraction of entries of x within α standard deviations of $\mathbf{avg}(x)$ is at least $1 - 1/\alpha^2$ (for $\alpha > 1$).

As an example, consider a time series of return on an investment, with a mean return of 8%, and a risk (standard deviation) 3%. By the Chebyshev inequality, the fraction of periods with a loss (*i.e.*, $x_i \leq 0$) is no more than $(3/8)^2 = 14.1\%$. (In fact, the fraction of periods when the return is either a loss, $x_i \leq 0$, or very good, $x_i \geq 16\%$, is together no more than 14.1%.)

Properties of standard deviation.

- *Adding a constant.* For any vector x and any number a, we have $\mathbf{std}(x+a\mathbf{1}) = \mathbf{std}(x)$. Adding a constant to every entry of a vector does not change its standard deviation.

- *Multiplying by a scalar.* For any vector x and any number a, we have $\mathbf{std}(ax) = |a|\,\mathbf{std}(x)$. Multiplying a vector by a scalar multiplies the standard deviation by the absolute value of the scalar.

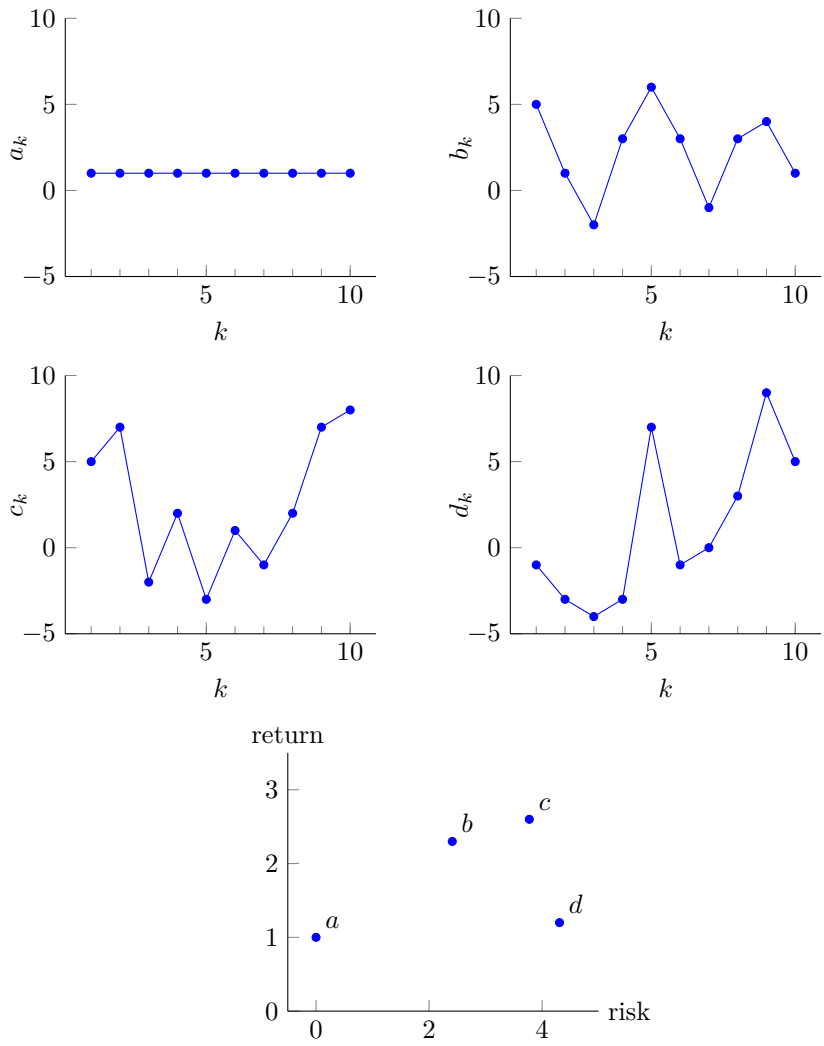

Figure 3.4 The vectors a, b, c, d represent time series of returns on investments over 10 periods. The bottom plot shows the investments in a risk-return plane, with return defined as the average value and risk as the standard deviation of the corresponding vector.

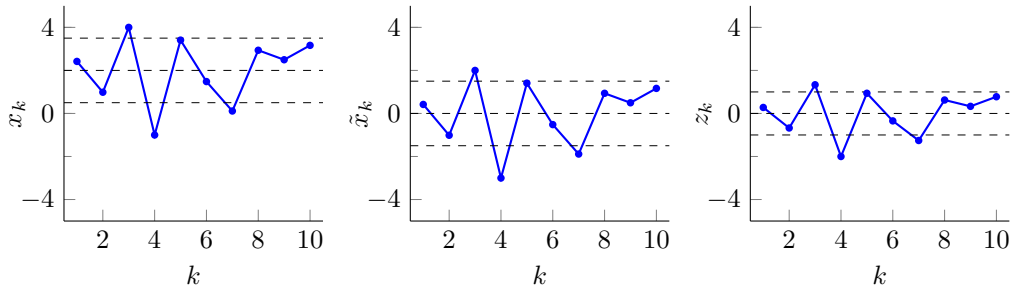

Figure 3.5 A 10-vector x, the de-meaned vector $\tilde{x} = x - \mathbf{avg}(x)\mathbf{1}$, and the standardized vector $z = (1/\mathbf{std}(x))\tilde{x}$. The horizontal dashed lines indicate the mean and the standard deviation of each vector. The middle line is the mean; the distance between the middle line and the other two is the standard deviation.

Standardization. For any vector x, we refer to $\tilde{x} = x - \mathbf{avg}(x)\mathbf{1}$ as the de-meaned version of x, since it has average or mean value zero. If we then divide by the RMS value of $\tilde{x}$ (which is the standard deviation of x), we obtain the vector

$$z = \frac{1}{\mathbf{std}(x)}(x - \mathbf{avg}(x)\mathbf{1}).$$

This vector is called the *standardized* version of x. It has mean zero, and standard deviation one. Its entries are sometimes called the *z-scores* associated with the original entries of x. For example, $z_4 = 1.4$ means that x_4 is 1.4 standard deviations above the mean of the entries of x. Figure 3.5 shows an example.

The standardized values for a vector give a simple way to interpret the original values in the vectors. For example, if an n-vector x gives the values of some medical test of n patients admitted to a hospital, the standardized values or z-scores tell us how high or low, compared to the population, that patient's value is. A value $z_6 = -3.2$, for example, means that patient 6 has a very low value of the measurement; whereas $z_{22} = 0.3$ says that patient 22's value is quite close to the average value.

3.4 Angle

Cauchy–Schwarz inequality. An important inequality that relates norms and inner products is the *Cauchy–Schwarz inequality*:

$$|a^T b| \leq \|a\| \, \|b\|$$

for any n-vectors a and b. Written out in terms of the entries, this is

$$|a_1 b_1 + \cdots + a_n b_n| \leq \left(a_1^2 + \cdots + a_n^2\right)^{1/2} \left(b_1^2 + \cdots + b_n^2\right)^{1/2},$$

which looks more intimidating. This inequality is attributed to the mathematician Augustin-Louis Cauchy; Hermann Schwarz gave the derivation given below.

The Cauchy–Schwarz inequality can be shown as follows. The inequality clearly holds if $a = 0$ or $b = 0$ (in this case, both sides of the inequality are zero). So we suppose now that $a \neq 0$, $b \neq 0$, and define $\alpha = \|a\|$, $\beta = \|b\|$. We observe that

$$\begin{aligned} 0 &\leq \|\beta a - \alpha b\|^2 \\ &= \|\beta a\|^2 - 2(\beta a)^T(\alpha b) + \|\alpha b\|^2 \\ &= \beta^2\|a\|^2 - 2\beta\alpha(a^T b) + \alpha^2\|b\|^2 \\ &= \|b\|^2\|a\|^2 - 2\|b\|\|a\|(a^T b) + \|a\|^2\|b\|^2 \\ &= 2\|a\|^2\|b\|^2 - 2\|a\|\,\|b\|(a^T b). \end{aligned}$$

Dividing by $2\|a\|\,\|b\|$ yields $a^T b \leq \|a\|\,\|b\|$. Applying this inequality to $-a$ and b we obtain $-a^T b \leq \|a\|\,\|b\|$. Putting these two inequalities together we get the Cauchy–Schwarz inequality, $|a^T b| \leq \|a\|\,\|b\|$.

This argument also reveals the conditions on a and b under which they satisfy the Cauchy–Schwarz inequality with equality. This occurs only if $\|\beta a - \alpha b\| = 0$, *i.e.*, $\beta a = \alpha b$. This means that each vector is a scalar multiple of the other (in the case when they are nonzero). This statement remains true when either a or b is zero. So the Cauchy–Schwarz inequality holds with equality when one of the vectors is a multiple of the other; in all other cases, it holds with strict inequality.

Verification of triangle inequality. We can use the Cauchy–Schwarz inequality to verify the triangle inequality. Let a and b be any vectors. Then

$$\begin{aligned} \|a + b\|^2 &= \|a\|^2 + 2a^T b + \|b\|^2 \\ &\leq \|a\|^2 + 2\|a\|\|b\| + \|b\|^2 \\ &= (\|a\| + \|b\|)^2, \end{aligned}$$

where we used the Cauchy–Schwarz inequality in the second line. Taking the squareroot we get the triangle inequality, $\|a + b\| \leq \|a\| + \|b\|$.

Angle between vectors. The *angle* between two nonzero vectors a, b is defined as

$$\theta = \arccos\left(\frac{a^T b}{\|a\|\,\|b\|}\right)$$

where arccos denotes the inverse cosine, normalized to lie in the interval $[0, \pi]$. In other words, we define θ as the unique number between 0 and π that satisfies

$$a^T b = \|a\|\,\|b\| \cos\theta.$$

The angle between a and b is written as $\angle(a, b)$, and is sometimes expressed in degrees. (The default angle unit is *radians*; 360° is 2π radians.) For example, $\angle(a, b) = 60^\circ$ means $\angle(a, b) = \pi/3$, *i.e.*, $a^T b = (1/2)\|a\|\|b\|$.

The angle coincides with the usual notion of angle between vectors, when they have dimension two or three, and they are thought of as displacements from a

common point. For example, the angle between the vectors $a = (1, 2, -1)$ and $b = (2, 0, -3)$ is

$$\arccos\left(\frac{5}{\sqrt{6}\sqrt{13}}\right) = \arccos(0.5661) = 0.9690 = 55.52^\circ$$

(to 4 digits). But the definition of angle is more general; we can refer to the angle between two vectors with dimension 100.

The angle is a symmetric function of a and b: We have $\angle(a, b) = \angle(b, a)$. The angle is not affected by scaling each of the vectors by a positive scalar: We have, for any vectors a and b, and any positive numbers α and β,

$$\angle(\alpha a, \beta b) = \angle(a, b).$$

Acute and obtuse angles. Angles are classified according to the sign of $a^T b$. Suppose a and b are nonzero vectors of the same size.

- If the angle is $\pi/2 = 90^\circ$, *i.e.*, $a^T b = 0$, the vectors are said to be *orthogonal*. We write $a \perp b$ if a and b are orthogonal. (By convention, we also say that a zero vector is orthogonal to any vector.)
- If the angle is zero, which means $a^T b = \|a\|\|b\|$, the vectors are *aligned*. Each vector is a positive multiple of the other.
- If the angle is $\pi = 180^\circ$, which means $a^T b = -\|a\|\,\|b\|$, the vectors are *anti-aligned*. Each vector is a negative multiple of the other.
- If $\angle(a, b) < \pi/2 = 90^\circ$, the vectors are said to make an *acute angle*. This is the same as $a^T b > 0$, *i.e.*, the vectors have positive inner product.
- If $\angle(a, b) > \pi/2 = 90^\circ$, the vectors are said to make an *obtuse angle*. This is the same as $a^T b < 0$, *i.e.*, the vectors have negative inner product.

These definitions are illustrated in figure 3.6.

Examples.

- *Spherical distance.* Suppose a and b are 3-vectors that represent two points that lie on a sphere of radius R (for example, locations on earth). The spherical distance between them, measured along the sphere, is given by $R\angle(a, b)$. This is illustrated in figure 3.7.
- *Document similarity via angles.* If n-vectors x and y represent the word counts for two documents, their angle $\angle(x, y)$ can be used as a measure of document dissimilarity. (When using angle to measure document dissimilarity, either word counts or histograms can be used; they produce the same result.) As an example, table 3.2 gives the angles in degrees between the word histograms in the example at the end of §3.2.